

# HTTP API manuál pro zařízení 2N

**2N**

## Obsah:

- 1. Úvod
  - 1.1 Revize dokumentu
- 2. Popis protokolu HTTP API
  - 2.1 Metody HTTP protokolu
  - 2.2 Parametry požadavků
  - 2.3 Odpovědi na požadavky
- 3. Zabezpečení služeb HTTP API
- 4. Uživatelské účty
- 5. Přehled funkcí HTTP API
  - 5.1 api system
    - 5.1.1 api system info
    - 5.1.2 api system status
    - 5.1.3 api system restart
    - 5.1.4 api system caps
    - 5.1.5 api system time
    - 5.1.6 api system time set
    - 5.1.7 api system timezone
    - 5.1.8 api system timezone caps
  - 5.2 api firmware
    - 5.2.1 api firmware
    - 5.2.2 api firmware apply
    - 5.2.3 api firmware reject
  - 5.3 api config
    - 5.3.1 api config
    - 5.3.2 api config factoryreset
    - 5.3.3 api config holidays
  - 5.4 api switch
    - 5.4.1 api switch caps
    - 5.4.2 api switch status
    - 5.4.3 api switch ctrl
  - 5.5 api io
    - 5.5.1 api io caps
    - 5.5.2 api io status
    - 5.5.3 api io ctrl
  - 5.6 api phone
    - 5.6.1 api phone status
    - 5.6.2 api phone callog
    - 5.6.3 api phone config
  - 5.7 api call
    - 5.7.1 api call status
    - 5.7.2 api call dial
    - 5.7.3 api call answer

- 5.7.4 api call hangup
- 5.8 api camera
  - 5.8.1 api camera caps
  - 5.8.2 api camera snapshot
- 5.9 api display
  - 5.9.1 api display caps
  - 5.9.2 api display image
    - 5.9.2.1 Příklady api display image
- 5.10 api log
  - 5.10.1 api log caps
  - 5.10.2 api log subscribe
  - 5.10.3 api log unsubscribe
  - 5.10.4 api log pull
- 5.11 api audio
  - 5.11.1 api audio test
- 5.12 api email
  - 5.12.1 api email send
- 5.13 api pcap
  - 5.13.1 api pcap
  - 5.13.2 api pcap restart
  - 5.13.3 api pcap stop
  - 5.13.4 api pcap live
  - 5.13.5 api pcap live stop
  - 5.13.6 api pcap live stats
- 5.14 api dir
  - 5.14.1 api dir template
  - 5.14.2 api dir create
  - 5.14.3 api dir update
  - 5.14.4 api dir delete
  - 5.14.5 api dir get
  - 5.14.6 api dir query
- 5.15 api mobilekey
  - 5.15.1 api mobilekey config
- 5.16 api lpr
  - 5.16.1 api lpr licenseplate
  - 5.16.2 api lpr image
- 5.17 api accesspoint
  - 5.17.1 api accesspoint blocking ctrl
  - 5.17.2 api accesspoint blocking status
  - 5.17.3 api accesspoint grantaccess
- 5.18 api lift
  - 5.18.1 api lift grantaccess
- 5.19 api automation
  - 5.19.1 api automation trigger

- [5.20 api cert](#)
  - [5.20.1 api cert ca](#)
  - [5.20.2 api cert user](#)

## 1. Úvod

**HTTP API** je aplikační rozhraní pro ovládání vybraných funkcí zařízení pomocí **HTTP** protokolu. Toto rozhraní umožňuje jednoduše integrovat zařízení 2N s produkty třetích stran, např. systémy domácí automatizace, zabezpečovací a monitorovací systémy budov apod.

**HTTP API** je podle funkce rozděleno do následujících služeb:

- **System API** – umožňuje změny konfigurace, získání stavu a upgrade zařízení.
- **Switch API** – umožňuje řízení a sledování stavu spínačů, např. otevírání dveřních zámků apod.
- **I/O API** – umožňuje řízení a sledování logických vstupů a výstupů zařízení.
- **Audio API** – umožňuje řízení přehrávání zvuků a monitorování mikrofonu zařízení.
- **Camera API** – umožňuje řízení a sledování obrazu z kamery.
- **Display API** – umožňuje řízení displeje a zobrazování uživatelských informací na displeji.
- **E-mail API** – umožňuje ze zařízení odesílat uživatelské e-maily.
- **Phone/Call API** – umožňuje řízení a sledování příchozích a odchozích hovorů.
- **Logging API** – umožňuje vyčítat zaznamenané události zařízení

Pro každou službu lze nastavit transportní protokol (**HTTP** nebo **HTTPS**) a způsob autentizace (**žádná**, **Basic** nebo **Digest**). V konfiguraci **HTTP API** lze vytvořit až pět uživatelských účtů (s vlastním jménem a heslem) s možností detailního řízení přístupu k jednotlivým službám a funkcím.

**HTTP API** se konfiguruje pomocí konfiguračního webového rozhraní zařízení v záložce **Služby / HTTP API**. Zde lze povolovat a konfigurovat jednotlivé služby a nastavovat parametry uživatelských účtů. Pro demonstraci a odzkoušení funkce **HTTP API** slouží speciální nástroj integrovaný v **HTTP** serveru zařízení dostupný na adrese **http(s)://ip\_adresa\_zařízení/apitest.html**.

**⚠ Upozornění****Varování**

Za účelem dosažení plné funkčnosti a zaručených výkonů důrazně doporučujeme vždy již při instalaci ověřit aktuálnost používané verze produktu či zařízení. Zákazník tímto bere na vědomí, že produkt či zařízení může dosahovat zaručených výkonů a být plně funkční dle propozic výrobce pouze v případě, je-li používána nejnovější verze produktu či zařízení, která byla otestována na plnou interoperabilitu a která nebyla výrobcem označena jako nekompatibilní s určitými verzemi jiných produktů, a to pouze v souladu s pokyny, návodem či doporučením výrobce a pouze ve spojení s vyhovujícími produkty a zařízeními jiných výrobců. Nejnovější verze jsou dostupné na internetových stránkách [https://www.2n.com/cs\\_CZ/](https://www.2n.com/cs_CZ/), popř. jednotlivá zařízení podle svých technických možností umožňují aktualizaci v konfiguračním rozhraní. Používá-li zákazník jinou než nejnovější verzi produktu či zařízení, popř. používá-li verzi, kterou výrobce označil za nekompatibilní s určitými verzemi jiných produktů, nebo používá-li zákazník produkt či zařízení v rozporu s pokyny, návodem či doporučením výrobce nebo ve spojení s nevhovujícími produkty či zařízeními jiných výrobců, je srozuměn s veškerými případnými omezeními funkčnosti takového produktu či zařízení a s důsledky s tím spojenými. Použitím jiné než nejnovější verze produktu či zařízení, popř. verze, kterou výrobce označil za nekompatibilní s určitými verzemi jiných produktů, nebo použitím produktu či zařízení v rozporu s pokyny, návodem či doporučením výrobce, popř. použitím s nevhovujícími produkty či zařízeními jiných výrobců, zákazník souhlasí s tím, že společnost 2N TELEKOMUNIKACE a.s. není odpovědná za jakékoli omezení funkčnosti takového produktu ani za újmu související s takovým případným omezením funkčnosti.

## 1.1 Revize dokumentu

## 1.1 Revize dokumentu

Verze	Popis změn
2.44	<ul style="list-style-type: none"> <li>• Podpora HTTP API zařízením <b>2N LiftIP 2.0</b>.</li> <li>• Doplněna nová funkce <b>api/system/timezone</b>.</li> <li>• Doplněna nová funkce <b>api/system/timezone/caps</b>.</li> <li>• Rozšířena funkce <b>api/system/time</b> o metodu <b>PUT</b> pro konfiguraci času.</li> </ul>
2.42	<ul style="list-style-type: none"> <li>• Doplněna nová funkce <b>api/cert/ca</b></li> <li>• Doplněna nová funkce <b>api/cert/user</b></li> </ul>
2.40	<ul style="list-style-type: none"> <li>• Rozšířena funkce <b>/api/lpr/licenseplate</b> o parametry <b>lprID</b> a <b>lprDir</b>.</li> </ul>

Verze	Popis změn
2.39	<ul style="list-style-type: none"> <li>Doplněna nová událost <b>DtmfSent</b>.</li> </ul>
2.38	<ul style="list-style-type: none"> <li>Událost <b>MotionDetection</b> rozšířena o parametr <b>ID</b> specifikující číslo profilu detekce pohybu ve webovém rozhraní.</li> </ul>
2.37	<ul style="list-style-type: none"> <li>Doplněna nová funkce <b>api/accesspoint/grantaccess</b>.</li> <li>Doplněna nová událost <b>ApiAccessRequested</b> generovaná při zaslání požadavku na <code>/api/accesspoint/grantaccess</code> s výsledkem "success" : true.</li> </ul>
2.36	<ul style="list-style-type: none"> <li>Rozšířena funkce <b>api/switch/status</b> o parametr <b>holdTimeout</b>.</li> <li>Rozšířena funkce <b>api/switch/ctrl</b> o parametr <b>timeout</b>.</li> <li>Doplněna funkce <b>api/phone/callog</b> pro stahování záznamu volání a mazání celého obsahu nebo jeho jednotlivých záznamů.</li> </ul>
2.35	<ul style="list-style-type: none"> <li>Doplněna funkce <b>api/system/time</b> pro získávání času zařízení.</li> <li>Doplněna funkce <b>api/system/time/set</b> pro nastavení času zařízení.</li> <li>Doplněna funkce <code>api/automation/trigger</code> pro aktivaci funkce automatizace události HttpTrigger.</li> <li>Doplněn nový parametr <b>users</b> do funkce <b>api/call/dial</b> pro volání na jednoho či více uživatelů.</li> </ul>
2.34	<ul style="list-style-type: none"> <li>Doplněna funkce <b>api/lift/grantaccess</b> pro aktivaci pater výtahu na základě autorizace v jiném zařízení.</li> </ul>
2.33	<ul style="list-style-type: none"> <li>Doplněna funkce <b>/api/pcap/live, api/pcap/live/stop, api/pcap/stats</b> pro ovládání zachytávání příchozích a odchozích paketů.</li> </ul>
2.32	<ul style="list-style-type: none"> <li>Doplněna funkce <b>/api/lpr/licenseplate</b> pro řízení přístupu pomocí rozpoznání registrační značky vozidla.</li> <li>Doplněna funkce <b>api/lpr/image</b> pro získávání obrázků přijatých z rozpoznávání registračních značek vozidla.</li> </ul>
2.31	<ul style="list-style-type: none"> <li>Doplněna funkce <b>/api/mobilekey/config</b> pro čtení a zapisování identifikátorů lokace a šifrovacích klíčů pro autentizaci přes Bluetooth.</li> </ul>

Verze	Popis změn
2.30	<ul style="list-style-type: none"> <li>• Parametr <b>apbBroken</b> odstraněn z události <b>AccessTaken</b>.</li> <li>• Parametr <b>apbBroken</b> přidán do události <b>UserAuthenticated</b>.</li> </ul>
2.29	<ul style="list-style-type: none"> <li>• Doplněna nová funkce <b>api system caps</b>.</li> <li>• Doplněna nová událost <b>CapabilitiesChanged</b>.</li> </ul>
2.28	<ul style="list-style-type: none"> <li>• Beze změny.</li> </ul>
2.27	<ul style="list-style-type: none"> <li>• Doplněny nové události: <b>LiftStatusChanged</b>, <b>LiftConfigChanged</b>, <b>LiftFloorEnabled</b>.</li> <li>• Doplněny funkce <b>api holidays</b>, <b>api config holidays</b>, <b>api dir template</b>, <b>api dir create</b>, <b>api dir update</b>, <b>api dir delete</b>, <b>api dir get</b>, <b>api dir query</b>.</li> </ul>
2.26	<ul style="list-style-type: none"> <li>• Doplněny nové události: <b>DtmfEntered</b>, <b>AccessTaken</b>, <b>ApLockStateChanged</b>, <b>RexActivated</b>.</li> </ul>
2.25	<ul style="list-style-type: none"> <li>• Beze změny.</li> </ul>
2.24	<ul style="list-style-type: none"> <li>• Změna nahrávání uživatelů do adresáře z důvodu zrušení pozic.</li> </ul>
2.23	<ul style="list-style-type: none"> <li>• Událost <b>UserRejected</b> rozšířena o parametr <b>switchDisabled</b>.</li> </ul>
2.22	<ul style="list-style-type: none"> <li>• Doplněny nové události <b>CardHeld</b>, <b>PairingStateChanged</b>, <b>SwitchesBlocked</b>, <b>FingerEntered</b>, <b>MobKeyEntered</b>, <b>DoorStateChanged</b>, <b>UserRejected</b>, <b>DisplayTouched</b>.</li> </ul>
2.21	<ul style="list-style-type: none"> <li>• Rozšířena funkce <b>api/display/image</b> (parametry <b>display</b>, <b>blob-image</b>, <b>blob-video</b>, <b>duration</b>, <b>repeat</b>) pro 2N<sup>®</sup> IP Verso.</li> <li>• Doplněny nové události <b>UserAuthenticated</b>, <b>SilentAlarm</b>, <b>AccessLimited</b>.</li> <li>• Doplněn nový parametr <b>timeSpan</b> do funkce <b>/api/email/send</b>.</li> </ul>



Verze	Popis změn
2.15	<ul style="list-style-type: none"> <li>• Doplněny nové události <b>TamperSwitchActivated</b>, <b>UnauthorizedDoorOpen</b>, <b>DoorOpenTooLong</b>, <b>LoginBlocked</b>.</li> <li>• Události rozšířeny o parametr <b>tzShift</b> udávající rozdíl mezi místním a UTC časem.</li> <li>• Funkce <b>email/send</b> rozšířena o možnost nastavení rozlišení odesílaných obrázků.</li> </ul>
2.14	<ul style="list-style-type: none"> <li>• Doplněny funkce <b>api/pcap</b>, <b>api/pcap/restart</b>, <b>api/pcap/stop</b> pro stahování a řízení záznamu síťového provozu.</li> <li>• Doplněna funkce <b>audio/test</b> pro spuštění automatického audio testu.</li> <li>• Doplněna funkce <b>email/send</b> pro odesílání e-mailu.</li> <li>• Doplněn nový parametr <b>response</b> do funkcí <b>/api/io/ctrl</b> a <b>/api/switch/ctrl</b>.</li> <li>• Funkce <b>/call/hangup</b> rozšířena o nový parametr <b>reason</b> umožňující zadat důvod ukončení hovoru.</li> <li>• Doplněny nové události <b>MotionDetected</b>, <b>NoiseDetected</b> a <b>SwitchStateChanged</b>.</li> <li>• Událost <b>CallStateChanged</b> rozšířena o parametr <b>reason</b> specifikující důvod ukončení hovoru.</li> </ul>
2.13	<ul style="list-style-type: none"> <li>• První verze dokumentu.</li> </ul>

## 2. Popis protokolu HTTP API

Všechny příkazy **HTTP API** jsou odesílány pomocí **HTTP** nebo **HTTPS** protokolu na adresu interkomu s absolutní cestou doplněným prefixem **/api**. Volba protokolu závisí na aktuálním nastavení interkomu v sekci **Služby / HTTP API**. Funkce **HTTP API** jsou rozděleny do služeb a u každé služby je možné nastavit požadovanou úroveň zabezpečení včetně požadavku na **TLS** spojení (tj. **HTTPS** protokol)

**Příklad:** Sepnutí spínače 1 <http://10.0.23.193/api/switch/ctrl?switch=1&action=on>

Absolutní cesta obsahuje název skupiny funkcí (systém, firmware, config, switch apod.) a název funkce samotné (caps, status, ctrl apod.).

Minimalistická varianta požadavku akceptovaná interkomem musí obsahovat řádek požadavku s metodou a absolutní cestou následovaný hlavičkou Host:

**Příklad:**

```
GET /api/system/info HTTP/1.1
Host: 10.0.23.193
HTTP Server interkomu odpoví zprávou:
HTTP/1.1 200 OK
Server: HIP2.10.0.19.2
Content-Type: application/json
Content-Length: 253
{
  "success" : true,
  "result" : {
    "variant" : "2N IP Vario",
    "serialNumber" : "08-1860-0035",
    "hwVersion" : "535v1",
    "swVersion" : "2.10.0.19.2",
    "buildType" : "beta",
    "deviceName" : "2N IP Vario"
  }
}
```

**V této kapitole dále naleznete:**

- [2.1 Metody HTTP protokolu](#)
- [2.2 Parametry požadavků](#)
- [2.3 Odpovědi na požadavky](#)

## 2.1 Metody HTTP protokolu

**2N IP interkom** využívá následující čtyři metody HTTP protokolu:

- **GET** – požadavky stahující obsah z interkomu nebo provádějící obecné příkazy
- **POST** – požadavky stahující obsah z interkomu nebo provádějící obecné příkazy
- **PUT** – požadavky pro nahrávání obsahu do interkomu
- **DELETE** – požadavky pro odstranění obsahu z interkomu

Metody **GET** a **POST** jsou z hlediska **HTTP API** ekvivalentní a liší se pouze způsobem předávání parametrů (viz následující kapitola). Metody **PUT** a **DELETE** se používají k manipulaci s velkými objekty jako je konfigurace, firmware, obrázky nebo zvukové soubory.

## 2.2 Parametry požadavků

Prakticky všechny funkce **HTTP API** mohou mít parametry. Parametry jsou pojmenované (switch, action, width, height, blob-image apod.) a jsou uvedeny v popisu příslušné funkce **HTTP API**. Parametry je možné v požadavku předávat třemi způsoby, které lze navzájem kombinovat:

1. v cestě požadavku (uri query, metody **GET**, **POST**, **PUT** a **DELETE**)
2. v obsahu zprávy (application/x-www-form-urlencoded, metody **POST** a **PUT**)
3. v obsahu zprávy (multipart/form-data, metody **POST** a **PUT**) – **RFC-1867**

V případě, že jednotlivé metody předávání parametrů se navzájem kombinují, může nastat situace, kdy je parametr v požadavku uveden vícekrát. V tomto případě se dává přednost poslednímu výskytu parametru.

Parametry funkcí **HTTP API** jsou dvou typů:

1. Parametr s jednoduchou hodnotou (switch, action apod.) může být předán pomocí všech třech výše uvedených metod. Tyto parametry nemají v názvu prefix blob-.
2. Parametr obsahující velká data (např. konfiguraci, firmware, obrázky apod.). Tyto parametry začínají vždy prefixem blob- a lze je předávat pouze pomocí poslední metody (multipart/form-data).

## 2.3 Odpovědi na požadavky

Odpovědi na požadavky jsou převážně ve formátu **JSON**. Výjimku tvoří pouze požadavky na stažení binárních dat (uživatelské zvuky, obrázky apod.) nebo konfiguraci interkomu v **XML**. Formát odpovědi lze rozlišit dle hlavičky Content-Type. Pro **JSON** jsou definovány tři základní typy odpovědí:

## Kladná odpověď bez parametrů

Tato odpověď je odesílána v případě úspěšného provedení požadavku u funkcí nevracejících žádné parametry. Tato odpověď je vždy kombinovaná se stavovým kódem **HTTP** protokolu **200 OK**.

```
{
  "success" : true,
}
```

## Kladná odpověď s parametry

Tato odpověď je odesílána v případě úspěšného provedení požadavku u funkcí vracejících doplňkové parametry. Položka **result** obsahuje další parametry odpovědi poplatné dané funkci. Tato odpověď je vždy kombinovaná se stavovým kódem **HTTP** protokolu **200 OK**.

```
{
  "success" : true,
  "result" : {
    ...
  }
}
```

## Záporná odpověď při chybě zpracování požadavku

Tento typ odpovědi je odeslán v případě, že při zpracování požadavku dojde k chybě. Odpověď nese kód chyby (parametr **code**), její textový popis (parametr **description**) a případně upřesnění chyby (parametr **param**). Tato odpověď může být kombinovaná se stavovým kódem **HTTP** protokolu **200 OK** nebo **401 Authorization Required**.

```
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "port",
    "description" : "invalid parameter value"
  }
}
```

V následující tabulce jsou vyjmenovány možné kódy chyb:

Kód	Popis	
1	function is not supported	Požadovaná funkce není na tomto modelu dostupná.
2	invalid request path	Absolutní cesta specifikovaná v <b>HTTP</b> požadavku neodpovídá žádné z funkcí <b>HTTP API</b> .
3	invalid request method	Použitá metoda <b>HTTP</b> protokolu není pro danou funkci platná.
4	function is disabled	Funkce (příslušná služba) není povolena. Službu je potřeba povolit na stránce konfiguračního rozhraní <b>Služby / HTTP API</b> .
7	invalid connection type	Je vyžadováno <b>HTTPS</b> připojení.
8	invalid authentication method	Použitá autentizační metoda není pro danou službu povolena. Tato chyba může nastat v případě, kdy pro službu je povolena pouze autentizační metoda Digest a klient se pokouší autentizovat pomocí metody Basic.
9	authorization required	Pro přístup ke službě je vyžadovaná autorizace uživatele. Tato chyba je odesílána společně stavovým kódem <b>HTTP</b> protokolu 401 Authorization Required.
10	insufficient user privileges	Autentizovaný uživatel nemá dostatečná privilegia pro provedení funkce.
11	missing mandatory parameter	V požadavku není uveden povinný parametr. Název chybějícího parametru je uveden v položce <b>param</b> .
12	invalid parameter value	Hodnota jednoho z parametrů požadavku není platná. Název neplatného parametru je uveden v položce <b>param</b> .

Kód	Popis	
13	parameter data too big	Data parametru překračují maximální povolenou velikost. Název chybného parametru je uveden v položce <b>param</b> .
14	unspecified processing error	Nastala nespecifikovaná chyba při zpracování požadavku.
15	no data available	Server nemá k dispozici požadovaná data.
17	parameter shouldn't be present	Kolize parametrů (není možné napsat danou kombinaci parametrů).
18	request is rejected	Požadavek nelze nyní zpracovat a byl zařízením odmítnut.
19	file version is lower than minimum	Zadaná verze souboru je nižší, než je požadováno.

### 3. Zabezpečení služeb HTTP API

V konfiguračním webovém rozhraní **2N IP** interkomu na stránce **Služby / HTTP API** lze nastavovat úroveň zabezpečení jednotlivých služeb **HTTP API**. Služby lze vypnout, zapnout nebo nastavit požadovaný komunikační protokol a způsob autentizace uživatelů.

SLUŽBA	POVOLENO	TYP PŘIPOJENÍ	AUTENTIZACE
System API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Access Control API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Switch API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
I/O API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Audio API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Camera API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Display API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
E-Mail API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Phone/Call API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Logging API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼
Automation API	<input checked="" type="checkbox"/>	Zabezpečené (TLS) ▼	Digest ▼

U každé služby lze nezávisle nastavit požadovaný transportní protokol:

- **HTTP** – požadavky mohou být odesílány pomocí **HTTP** nebo **HTTPS** protokolu. Oba protokoly jsou povoleny a úroveň zabezpečení definuje klient použitým protokolem.
- **HTTPS** – požadavky musí být odesílány pomocí **HTTPS** protokolu a požadavky odesílané pomocí nezabezpečeného **HTTP** protokolu jsou interkomem odmítány. **HTTPS** protokol zajišťuje, že případný útočník nemůže číst obsah zpráv odesílaných a přijímaných zpráv.

U každé služby lze nastavit vyžadovaný způsob autentizace požadavků odesílaných na interkom. Pokud autentizace není provedena, požadavek je odmítnut. Požadavky jsou autentizovány pomocí standardního autentizačního protokolu popsáno v **RFC-2617**. Je možné volit tyto tři metody autentizace:

- **Žádná** – služba nevyžaduje žádnou autentizaci. Služba je v tomto případě v lokální síti zcela nechráněná.

- **Basic** – služba vyžaduje autentizaci Basic podle **RFC-2617**. Služba v tomto případě vyžaduje heslo, to je však odesíláno v otevřeném formátu. Doporučujeme tuto volbu kombinovat s **HTTPS** protokolem, pokud je to možné.
- **Digest** – služba vyžaduje autentizaci Digest podle **RFC-2617**. Tato varianta je výchozí a z výše uvedených metod nejbezpečnější.

Pro maximální bezpečnost a odolnost proti zneužití doporučujeme u všech služeb využívat kombinaci **HTTPS + Digest**. V případě, že druhá strana komunikující s interkomem tuto kombinaci nepodporuje, lze konkrétní službě udělit výjimku a úroveň zabezpečení snížit.



## 4. Uživatelské účty

**2N IP interkom** umožňuje spravovat až pět uživatelských účtů určených pro přístup ke službám **HTTP API**. Součástí uživatelského účtu je jméno a heslo uživatele a tabulka přístupových práv uživatele k jednotlivým službám **HTTP API**.

Účet povolen

Nastavení uživatele ▾

Jméno uživatele

Heslo

Uživatelská práva ▾

POPIS	SLEDOVÁNÍ	ŘÍZENÍ
System	<input type="checkbox"/>	<input type="checkbox"/>
Telefon/hovory	<input type="checkbox"/>	<input type="checkbox"/>
Správa přístupu	<input type="checkbox"/>	<input type="checkbox"/>
Vstupy a výstupy	<input type="checkbox"/>	<input type="checkbox"/>
Spínače		<input type="checkbox"/>
Audio		<input type="checkbox"/>
Kamera	<input type="checkbox"/>	
Displej		<input type="checkbox"/>
E-mail		<input type="checkbox"/>
UID (karty a Wiegand)	<input type="checkbox"/>	
Klávesnice	<input type="checkbox"/>	
Přístup k automatizaci		<input type="checkbox"/>

Pomocí tabulky přístupových práv lze řídit privilegia uživatelského účtu k jednotlivým službám.

## 5. Přehled funkcí HTTP API

V tabulce níže je souhrn všech dostupných funkcí **HTTP API**. Tabulka obsahuje následující informace:

- absolutní cesta **HTTP** požadavku
- podporované **HTTP** metody
- služba, ve které se funkce nachází
- vyžadovaná uživatelská práva uživatele (v případě, že se využívá autentizace)
- od verze FW 2.35 není využití vybrané funkce podmíněno licencí (tzn. funkce je dostupná bez nutnosti vložení licenčního klíče)

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/automation/trigger	GET	Automation	Přístup k automatizaci
/api/accesspoint/blocking/ctrl	GET/POST	Access Control	Správa přístupu – řízení
/api/accesspoint/blocking/status	GET/POST	Access Control	Správa přístupu – sledování
/api/accesspoint/grantaccess	GET/POST	Access Control	Správa přístupu – řízení
/api/audio/test	GET/POST	Audio	Audio – řízení
/api/call/answer	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/dial	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/hangup	GET/POST	Phone/Call	Telefon/hovory – řízení
/api/call/status	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/camera/caps	GET/POST	Camera	Kamera – sledování
/api/camera/snapshot	GET/POST	Camera	Kamera – sledování
/api/config	GET/POST/PUT	System	Systém – řízení
/api/config/factoryreset	GET/POST	System	Systém – řízení
/api/config/holidays	GET/PUT	System	Systém – řízení
/api/dir/create	PUT	System	Systém – řízení
/api/dir/delete	PUT	System	Systém – řízení
/api/dir/get	POST	System	Systém – řízení
/api/dir/query	POST	System	Systém – řízení
/api/dir/template	GET/POST	System	Systém – řízení
/api/dir/update	PUT	System	Systém – řízení

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/display/caps	GET/POST	Display	Displej – řízení
/api/display/image	PUT/DELETE	Display	Displej – řízení
/api/email/send	GET/POST	E-mail	E-mail – řízení
/api/firmware	PUT	System	Systém – řízení
/api/firmware/apply	GET/POST	System	Systém – řízení
/api/firmware/reject	GET/POST	System	Systém – řízení
/api/holidays	GET/PUT	System	Systém – řízení
/api/io/caps	GET/POST	I/O	Vstupy a výstupy – sledování
/api/io/ctrl	GET/POST	I/O	Vstupy a výstupy – řízení
/api/io/status	GET/POST	I/O	Vstupy a výstupy – sledování
/api/lift/grantaccess	GET/POST	Access Control	Správa přístupu – řízení
/api/log/caps	GET/POST	Logging	–
/api/log/pull	GET/POST	Logging	–
/api/log/subscribe	GET/POST	Logging	*
/api/log/unsubscribe	GET/POST	Logging	*
/api/lpr/image	GET/POST	Access Control	Správa přístupu – sledování
/api/lpr/licenseplate	POST	Access Control	Správa přístupu – řízení
/api/mobilekey/config	GET/PUT	Access Control	Správa přístupu – sledování
/api/pcap	GET/POST	System	Systém – řízení
/api/pcap/live	GET/POST	System	Systém – řízení

Absolutní cesta	Metoda	Služba	Potřebná uživatelská práva
/api/pcap/live/stats	GET/POST	System	Systém – řízení
/api/pcap/live/stop	GET/POST	System	Systém – řízení
/api/pcap/restart	GET/POST	System	Systém – řízení
/api/pcap/stop	GET/POST	System	Systém – řízení
/api/phone/callog	DELETE	Phone/Call	Telefon/hovory – řízení
/api/phone/callog	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/phone/status	GET/POST	Phone/Call	Telefon/hovory – sledování
/api/switch/caps	GET/POST	Switch	Spínače – sledování
/api/switch/ctrl	GET/POST	Switch	Spínače – řízení
/api/switch/status	GET/POST	Switch	Spínače – řízení
/api/system/caps	GET	System	Systém – sledování
/api/system/info	GET/POST	System	–
/api/system/restart	GET/POST	System	Systém – řízení
/api/system/status	GET/POST	System	Systém – řízení
/api/system/time	GET/POST/ PUT	System	Systém – sledování/řízení
/api/system/time/set	GET/POST	System	Systém – řízení
/api/system/timezone	GET/PUT	System	Systém – sledování/řízení
/api/system/timezone/caps	GET	Syste,	Systém – sledování
/api/cert/ca	GET/PUT/ DELETE	System	Systém – řízení
/api/cert/user	GET/PUT/ DELETE	System	Systém – řízení

**V této kapitole dále naleznete:**

- 5.1 api system
  - 5.1.1 api system info
  - 5.1.2 api system status
  - 5.1.3 api system restart
  - 5.1.4 api system caps
  - 5.1.5 api system time
  - 5.1.6 api system time set
  - 5.1.7 api system timezone
  - 5.1.8 api system timezone caps
- 5.2 api firmware
  - 5.2.1 api firmware
  - 5.2.2 api firmware apply
  - 5.2.3 api firmware reject
- 5.3 api config
  - 5.3.1 api config
  - 5.3.2 api config factoryreset
  - 5.3.3 api config holidays
- 5.4 api switch
  - 5.4.1 api switch caps
  - 5.4.2 api switch status
  - 5.4.3 api switch ctrl
- 5.5 api io
  - 5.5.1 api io caps
  - 5.5.2 api io status
  - 5.5.3 api io ctrl
- 5.6 api phone
  - 5.6.1 api phone status
  - 5.6.2 api phone calllog
  - 5.6.3 api phone config
- 5.7 api call
  - 5.7.1 api call status
  - 5.7.2 api call dial
  - 5.7.3 api call answer
  - 5.7.4 api call hangup
- 5.8 api camera
  - 5.8.1 api camera caps
  - 5.8.2 api camera snapshot
- 5.9 api display
  - 5.9.1 api display caps
  - 5.9.2 api display image
    - 5.9.2.1 Příklady api display image
- 5.10 api log
  - 5.10.1 api log caps
  - 5.10.2 api log subscribe

- 5.10.3 api log unsubscribe
- 5.10.4 api log pull
- 5.11 api audio
  - 5.11.1 api audio test
- 5.12 api email
  - 5.12.1 api email send
- 5.13 api pcap
  - 5.13.1 api pcap
  - 5.13.2 api pcap restart
  - 5.13.3 api pcap stop
  - 5.13.4 api pcap live
  - 5.13.5 api pcap live stop
  - 5.13.6 api pcap live stats
- 5.14 api dir
  - 5.14.1 api dir template
  - 5.14.2 api dir create
  - 5.14.3 api dir update
  - 5.14.4 api dir delete
  - 5.14.5 api dir get
  - 5.14.6 api dir query
- 5.15 api mobilekey
  - 5.15.1 api mobilekey config
- 5.16 api lpr
  - 5.16.1 api lpr licenseplate
  - 5.16.2 api lpr image
- 5.17 api accesspoint
  - 5.17.1 api accesspoint blocking ctrl
  - 5.17.2 api accesspoint blocking status
  - 5.17.3 api accesspoint grantaccess
- 5.18 api lift
  - 5.18.1 api lift grantaccess
- 5.19 api automation
  - 5.19.1 api automation trigger
- 5.20 api cert
  - 5.20.1 api cert ca
  - 5.20.2 api cert user

### 5.1 api system

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/system**.

- 5.1.1 api system info
- 5.1.2 api system status
- 5.1.3 api system restart

- [5.1.4 api system caps](#)
- [5.1.5 api system time](#)
- [5.1.6 api system time set](#)
- [5.1.7 api system timezone](#)
- [5.1.8 api system timezone caps](#)

### 5.1.1 api system info

Funkce **/api/system/info** slouží k získání základních informací o zařízení, jako je typ, výrobní číslo, verze firmware apod. Funkce je dostupná na všech typech zařízení bez ohledu na nastavená přístupová práva.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

Parametr	Popis
<b>devType</b>	Interní identifikátor zařízení.
<b>variant</b>	Název modelu (varianty) zařízení
<b>variantId</b>	Interní číselný identifikátor produktu
<b>customerId</b>	Interní číselný identifikátor
<b>serialNumber</b>	Sériové (výrobní) číslo zařízení
<b>macAddr</b>	Síťový identifikátor zařízení
<b>hwVersion</b>	Verze hardware
<b>swVersion</b>	Verze firmware
<b>buildType</b>	Typ sestavení firmware (alpha, beta, release)
<b>firmwarePackage</b>	Označení FW balíčku



Parametr	Popis
<b>deviceName</b>	Název zařízení nastavení v konfiguračním rozhraní v sekci <b>Služby / Web Server</b>

 **Upozornění**

- Od verze 2.33.2 se mění rozsah hodnot pro klíč "buildType", pro oficiální verzi bude hodnota obsahovat řetězec "release". Rozsah hodnot pro klíč "buildType" je do verze 2.32.1 pro oficiální verzi prázdný.

**Příklad:**

```
GET /api/system/info
{
  "success" : true,
  "result" : {
    "devType" : "2-14-0-0",
    "variant" : "2N IP Verso",
    "variantId" : 14,
    "customerId" : 0,
    "serialNumber" : "00-0000-0005",
    "macAddr" : "FC-1E-B3-00-00-05",
    "hwVersion" : "570v1",
    "swVersion" : "2.35.0.45.0",
    "buildType" : "dev",
    "firmwarePackage" : "verso",
    "deviceName" : "2N IP Verso"
  }
}
```

## 5.1.2 api system status

Funkce **/api/system/status** vrací aktuální stav interkomu.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje aktuální stav zařízení:

Parametr	Popis
<b>systemTime</b>	Reálný čas v zařízení v sekundách od 00:00 1.1.1970 (unix time)
<b>upTime</b>	Doba chodu zařízení od posledního restartu v sekundách.

**Příklad:**

```
GET /api/system/status
{
  "success" : true,
  "result" : {
    "systemTime" : 1418225091,
    "upTime" : 190524
  }
}
```

### 5.1.3 api system restart

Funkce **/api/system/restart** provede restart interkomu.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby měl uživatel přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/system/restart
{
  "success" : true
}
```

### 5.1.4 api system caps

Funkce **/api/system/caps** slouží k zasílání informací do **2N<sup>®</sup> Access Commanderu** o změně seznamu dostupných funkcí zařízení.

Funkce je součástí služby **System API** a v případě použití autentizace je nutné, aby měl uživatel přiřazené privilegium **Systém**.

Pro tuto funkci lze použít metodu **GET**.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

```
{  "success":true,
  "result":{
    "options":{
      "codecG722":"active,licensed",
      "codecG729":"active",
      "codecL16":"active",
      "audioLoopTest":"active,licensed",
      "noiseDetection":"active,licensed",
      "userSounds":"active,licensed",
      "adaptiveVolume":"active",
      "antiHowling":"active",
      "keyBeep":"active",
      "camera":"active",
      "video":"active",
      "cameraPtz":"active,licensed",
      "motionDetection":"active,licensed",
      "encH264":"active",
      "encH263":"active",
      "encMpeg4":"active",
      "encJpeg":"active",
      "decH264":"active",
      "phone":"active",
      "phoneVideo":"active",
      "phoneVideoOut":"active",
      "sips":"active,licensed",
      "srtp":"active,licensed",
      "callAnswerMode":"active",
      "doorOpenCallback":"active",
      "rtspServer":"active,licensed",
      "rtspClient":"active,licensed",
      "audioMulticast":"active",
      "smtpClient":"active,licensed",
      "ftpClient":"active,licensed",
      "onvif":"active,licensed",
      "snmp":"active,licensed",
      "tr069":"active,licensed",
      "knocker":"active",
      "my2n":"active",
      "informacast":"active,licensed",
      "autoProv":"active,licensed",
      "httpApi":"active,licensed",
      "eap":"active,licensed",
      "eapMd5":"active,licensed",
      "eapTls":"active,licensed",
      "vpn":"active",
      "userVpn":"active",
      "rioManager":"active,licensed",
      "siteChannel":"active",
      "localCalls":"active",
      "switches":"active",
      "advancedSwitches":"active,licensed",
      "switchUserCodes":"active",
```

```

    "securedInput":"active",
    "rexInput":"active",
    "tamperInput":"active",
    "doorSensor":"active",
    "keypad":"active",
    "buttons":"active",
    "liftControl":"active,licensed",
    "limitFailedAccess":"active,licensed",
    "silentAlarm":"active,licensed",
    "scrambleKeypad":"active,licensed",
    "tamperBlockSwitch":"active,licensed",
    "antiPassback":"active,licensed",
    "dir":"active",
    "dirDeputy":"active",
    "dirPhoto":"active",
    "automation":"active,licensed",
    "licDownload":"active",
    "profiles":"active",
    "licensing":"active",
    "accessControl":"active",
    "doorControl":"active",
    "nfc":"active,licensed",
    "vbus":"active",
    "vbusExtenders":"active",
    "cardReader":"active",
    "fpReader":"active",
    "bleReader":"active",
    "wiegand":"active",
    "powerManager":"active",
    "audioInput":"active",
    "lightSensor":"active",
    "irLed":"active",
    "backlight":"active",
    "backlightDayNight":"active",
    "display":"active"
  }
}
}

```

### 5.1.5 api system time

Funkce **/api/system/time** slouží k získání o čase zařízení nebo k nastavení času.

Pro tuto funkci lze použít metodu **GET** k získání informací nebo **PUT** k nahrání konfigurace.

## Metoda GET

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **System (sledování)**.

Pro metodu **GET** nejsou definovány žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje reálný čas v zařízení v sekundách od 00:00 1.1.1970 (unix time).

```
GET /api/system/time
{
  "success" : true,
  "result" : {
    "utcTime" : 1639472172,
    "source" : "My2N",
    "automatic" : true,
  }
}
```

## Metoda PUT

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **System (řízení)**.

### ⚠ Upozornění

- Doporučujeme používat tento endpoint k nastavení času zařízení jen v případě, že parametr **Použít čas z internetu** je vypnutý. V případě, že je tento parametr zapnutý, čas bude přepsán časem z NTP serveru nebo z časové služby My2N.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
<b>automatic</b>	Automatické získávání času z NTP serveru Při vyplnění hodnoty true se další parametry nežadávají.
<b>utcTime</b>	Číslo = unixový čas, min. BuildTime, max. 2147483647
<b>server</b>	adresa NTP serveru (IPv4 adresa nebo doména)

Odpověď pro metodu **PUT** je ve formátu **application/json**.

```
PUT /api/system/time?automatic=0&server=pool.ntp.org&utcTime=1700829813
{
```

```
"success" : true,
}
```

### 5.1.6 api system time set

#### **Poznámka**

- Ve verzi firmwaru 2.44 byla tato funkce nahrazena metodou PUT požadavku [api system time](#).

Funkce **/api/system/time/set** slouží k nastavení času zařízení.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Parametry požadavku:

Parametr	Popis
<b>time</b>	Číslo = unixový čas, min. 0, max. 2147483647

Odpověď je ve formátu **application/json**.

#### **Příklad:**

```
{
  "success" : true
}
```

#### **Upozornění**

- Doporučujeme používat tento endpoint k nastavení času zařízení jen v případě, že parametr **Použít čas z internetu** je vypnutý. V případě, že je tento parametr zapnutý, čas bude přepsán časem z NTP serveru nebo z časové služby My2N.

### 5.1.7 api system timezone

Funkce **/api/system/timezone** slouží k získání informací o časové zóně zařízení nebo k nastavení časové zóny.

Pro tuto funkci lze použít metodu **GET** k získání informací nebo **PUT** k nahrání konfigurace.

## Metoda GET

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (sledování)**.

Pro metodu **GET** nejsou definovány žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje informace, zda je tato zóna nastavena automaticky a jaká časová zóna je nastavena. V případě, že je časová zóna zadaná manuální volbou, je v parametru **custom** zobrazeno pravidlo, podle kterého byla zóna nastavena.

### Příklad

```
GET /api/system/timezone
{
  "success": true,
  "result": {
    "automatic": true,
    "zone": "America/Los Angeles"
  }
}
nebo
{
  "success": true,
  "result": {
    "automatic": false
    "zone": "custom",
    "custom": "UTC-08:00"
  }
}
```

## Metoda PUT

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
<b>automatic</b>	0 nebo 1 Při vyplnění hodnoty true (1) se další parametry nezadávají.
<b>zone</b>	Definuje zónu. Lze zadat buď název zóny nebo "custom" pro umožnění manuálního zadání zóny. Seznam názvů zón je na endpointu <a href="#">5.1.8 api system timezone caps</a> .



Parametr	Popis
<b>custom</b>	Definuje zónu vlastním zapsáním ve formátu <i>UTC-08:00</i> , <i>UTC+03:00</i> apod. Je možné použít pouze při <code>zone="custom"</code> .

**Příklad**

```
PUT /api/system/timezone?automatic=0&zone=custom&custom=UTC-08:00
{
  "success" : true
}
```

**5.1.8 api system timezone caps**

Funkce **api/system/timezone/caps** vrací seznam dostupných časových zón.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json**.

```
GET /api/system/timezone/caps
{
  "success": true,
  "result": {
    "timezones": [
      "Africa/Abidjan",
      "Africa/Accra",
      "Africa/Bissau",
      "Africa/Monrovia",
      "Africa/Sao_Tome",
      "America/Danmarkshavn",
      "Antarctica/Troll",
      "Atlantic/Canary",
      "Atlantic/Faroe",
      "Atlantic/Madeira",
      "Atlantic/Reykjavik",
      "Etc/GMT",
      "Etc/UCT",
      "Etc/UTC",
      "Europe/Lisbon",
      "Europe/London",
      ...
    ]
  }
}
```

## 5.2 api firmware

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/firmware**.

- [5.2.1 api firmware](#)
- [5.2.2 api firmware apply](#)
- [5.2.3 api firmware reject](#)

### 5.2.1 api firmware

Funkce **api/firmware** umožňuje nahrát soubor s firmwarem za účelem upgradování nebo downgradování.

#### Metody

- PUT

#### Služby a privilegia

- Služby: Systém API
- Privilegia: Systém – řízení

#### Požadavek PUT

Požadavek obsahuje soubor ve formátu **multipart/form-data**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>blob-fw</b>	Ano	platný firmwarový binární soubor	-	Soubor s firmwarem

## Příklad požadavku PUT

```
http://192.168.1.1/api/firmware
```

## Odpověď na PUT

Odpověď je ve formátu **application/json**. Obsahuje klíče **success** a **result**. Hodnota **result** obsahuje různé klíče popsané v následující tabulce.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>fileId</b>	náhodný identifikátor (8 HEX znaků)	Obsahuje náhodný identifikátor nahraného souboru s firmwarem. Identifikátor musí být použit k potvrzení nahraného firmwaru pomocí funkce <b>api/firmware/apply</b> nebo k odmítnutí nahraného firmwaru pomocí funkce <b>api/firmware/reject</b> .
<b>version</b>	Řetězec s identifikací verze major.minor.patch.build.id	Obsahuje identifikaci verze nahraného firmwaru.
<b>downgrade</b>	true nebo false	Tento příznak je true, jestliže má nahraný firmware nižší verzi, než je právě v zařízení.
<b>note</b>	Řetězec s náhradními znaky (URL kódování)	Obsahuje aktualizací zprávu pro nahraný firmware (např. varování před velkými změnami).

## Příklad odpovědi na PUT

```
{ "success" : true, "result" : { "fileId" : "7d6adf16", "version" : "2.32.4.41.2",
"downgrade" : false, "note" : "EN:\r\nVER=2.20.0\r\nSome changes associated with the
downgrade to a lower version result in a loss of original settings in a certain part
of configuration.\r\n\r\n* All the cards installed in the **Directory \\/ Access
cards** menu are moved to the **Directory \\/ Users** menu as new users upon firmware
upgrade. Each user is automatically named as !Visitor #n, where n gives the user
number in the list. This change is irreversible upon downgrade.\r\n* Service cards
are now available in the **Hardware \\/ Card reader** menu.\r\n* All the user
access ... .. \u0043F\u00440\u0043E\u0044\u00438\u0043B\u00435\u0043C
\u0043F\u0043E\u0043B\u0044C\u00437\u0043E\u00432\u00430\u00442\u00435\u0043B\u0044F.
\r\n\r\n" } }
```

Vrátit se mohou tyto konkrétní chybové kódy:

- Error code 12
  - param = "blob-fw"
  - popis = "neplatná hodnota parametru"
  - Nahraný firmware neodpovídá požadavkům (neplatný soubor, firmware pro jiné zařízení...)
- Error code 19
  - popis = "verze souboru je nižší než požadovaná minimální verze"
  - Nahraný firmware má nižší verzi, než je pro zařízení povolena.

### Upozornění

Zařízení neodpovídá na požadavky na nahrání dalšího firmwaru, když v něm existuje předchozí verze. Nejprve s pomocí funkce **api/firmware/reject** odmítněte předchozí verzi firmwaru a pak nahrajte jinou. Nahraný firmware bude za 5 minut automaticky odmítnut, pokud se nepoužije.

## 5.2.2 api firmware apply

Funkce **api/firmware/apply** slouží k potvrzení nahraného firmwaru a provedení upgradu/downgradu zařízení.

### Metody

- GET
- POST

### Skupiny služeb a privilegií

- Služby: System API
- Privilegia: Systém – řízení

## Požadavek PUT

Požadavek obsahuje soubor v **URL**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>fileId</b>	Ano	Identifikátor firmwarového souboru	-	Tento parametr se musí shodovat s identifikátorem aktuálně nahraného firmwaru.

## Příklad požadavku GET nebo POST

```
http://192.168.1.1/api/firmware/apply?fileId=7d6adf16
```

## Odpověď na GET nebo POST

Odpověď je ve formátu **application/json**. Obsahuje klíč **success**. Je-li `success true`, firmware je použit a zařízení je upgradováno/downgradováno.

## Příklad odpovědi na GET nebo POST

```
{ "success" : true }
```

Vrátit se mohou tyto konkrétní chybové kódy:

- Error code 12
  - parametr = "fileId"
  - popis = "neplatný parametr"
  - Identifikátor souboru je neplatný (např. obsahuje jiné než hexadecimální znaky).
- Error code 14
  - popis = "nový firmware nenalezen"
  - Není nahrán žádný firmwarový soubor s tímto fileId.

### 5.2.3 api firmware reject

Funkce **api/firmware/reject** slouží k odmítnutí nahraného souboru s firmwarem.

#### Metody

- GET
- POST

#### Služby a privilegia

- Služby: System API
- Privilegia: Systém – Řízení

#### Požadavek PUT

Požadavek obsahuje soubor v **URL**.

Tabulka 1. Parametry požadavku

Parametr	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>fileId</b>	Ano	Identifikátor firmwarového souboru	-	Tento parametr se musí shodovat s identifikátorem aktuálně nahraného firmwaru.

#### Příklad požadavku GET nebo POST

```
http://192.168.1.1/api/firmware/reject?fileId=7d6adf16
```

#### Odpověď na GET nebo POST

Odpověď je ve formátu **application/json**. Obsahuje klíč **success**. Je-li success true, firmware je odmítnut a s pomocí funkce **api/firmware** je možno nahrát nový soubor.

## Příklad odpovědi na GET nebo POST

```
{ "success" : true }
```

Vrátit se mohou tyto konkrétní chybové kódy:

- Error code 12
  - parametr = "fileId"
  - popis = "neplatný parametr"
  - Identifikátor souboru je neplatný (např. obsahuje jiné než hexadecimální znaky).
- Error code 14
  - popis = "nový firmware nenalezen"
  - Není nahrán žádný firmwarový soubor s tímto fileId.

**⚠ Upozornění**

- Zařízení neodpovídá na požadavky **api/firmware** na nahrání dalšího firmwaru, když v něm existuje předchozí verze. Nejprve s pomocí funkce **api/firmware/reject** odmítněte předchozí verzi firmwaru a pak nahrajte jinou. Nahraný firmware bude za 5 minut automaticky odmítnut, pokud se nepoužije.

## 5.3 api config

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/config**.

- [5.3.1 api config](#)
- [5.3.2 api config factoryreset](#)
- [5.3.3 api config holidays](#)

### 5.3.1 api config

Funkce **/api/config** slouží k uploadu nebo downloadu konfigurace zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST** pro download konfigurace nebo metodu **PUT** pro upload konfigurace.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
<b>blob-cfg</b>	Povinný parametr obsahující konfiguraci zařízení (ve formátu XML).

Pro metody GET/POST nejsou definovány žádné parametry.

V případě downloadu konfigurace je odpověď ve formátu **application/xml** a obsahuje kompletní konfigurační soubor zařízení.

Funkce **/api/config** s použitím metody **PUT** provádí upload konfigurace se zpožděním cca 15 s, během tohoto intervalu se zařízení nesmí resetovat ani vypínat.



**Příklad:**

```

GET /api/config
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Product name: 2N IP Vario
    Serial number: 08-1860-0035
    Software version: 2.10.0.19.2
    Hardware version: 535v1
    Bootloader version: 2.10.0.19.1
    Display: No
    Card reader: No
-->
<DeviceDatabase Version="4">
<Network>
  <DhcpEnabled>1</DhcpEnabled>
  ...
  ...

```

V případě uploadu konfigurace je odpověď ve formátu **application/json** a neobsahuje žádné další parametry.

**Příklad:**

```

PUT /api/config
{
  "success" : true
}

```

**⚠ Upozornění**

- Po přechodu na verzi 2.24 dojde ke zrušení pozic uživatelů v adresáři. Při aktualizaci adresáře je nejprve potřeba stáhnout současnou konfiguraci, v této konfiguraci provést požadované změny a poté znovu nahrát.
- Při nedodržení postupu může dojít ke ztrátě dat.

### 5.3.2 api config factoryreset

Funce **/api/config/factoryreset** nastaví všechny parametry interkomu do výchozího stavu. Tato funkce je shodná se stejnojmennou funkcí webového konfiguračního rozhraní **Systém / Údržba – Výchozí nastavení**.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

Funkce **/api/config/factoryreset** nastavuje interkom do výchozího stavu se zpožděním cca 15 s, během tohoto intervalu se interkom nesmí resetovat ani vypínat.

**Příklad:**

```
GET /api/config/factoryreset
{
  "success" : true
}
```

### 5.3.3 api config holidays

Funkce **/api/config/holidays** slouží k vytváření/nastavování seznamu svátků.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **System (řízení)**.

Pro tuto funkci lze použít metodu **GET** nebo **PUT**.

Pro metodu **GET** nejsou definovány žádné parametry.

Parametry požadavku pro metodu **PUT**:

Parametr	Popis
<b>blob-json</b>	Povinný parametr obsahující definici svátků (JSON)

Odpověď pro metodu **GET** je ve formátu **application/json** a obsahuje pole svátků. Data jsou ve formátu DD/MM[/YYYY], přičemž rok se uvádí, když svátek platí jen pro zadaný rok.

GET /api/config/holidays

```
{
  "success" : true,
  "result" : { "dates": [ "01\01", "24\12", "01\04\2018" ] }
}
```

Formát JSON pro metodu **PUT** je stejný jako pro výsledek metody **GET**.

```
{ "dates": [ "01\01", "24\12", "01\04\2018" ] }
```

Odpověď pro metodu **PUT** je ve formátu **application/json** a neobsahuje žádné další parametry.

PUT /api/config/holidays

```
{ "success": true
}
```

## 5.4 api switch

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/switch**.

- [5.4.1 api switch caps](#)
- [5.4.2 api switch status](#)
- [5.4.3 api switch ctrl](#)

### 5.4.1 api switch caps

Funkce **/api/switch/caps** vrací aktuální nastavení a možnosti řízení spínačů. Funkce má volitelný parametr **switch**, který určuje spínač, jehož vlastnosti a nastavení se mají vrátit. Pokud parametr **switch** není uveden, funkce vrací stav všech spínačů.

Funkce je součástí služby **Switch** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Spínače (sledování)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>switch</b>	Volitelný parametr definující číslo spínače (obvykle 1 až 4).

Odpověď je ve formátu **application/json** a obsahuje seznam spínačů (pole **switches**) a jejich aktuální nastavení. V případě použití parametru **switch** obsahuje pole **switches** právě jednu položku.

Parametr	Popis
<b>switch</b>	ID spínače (1 až 4)

Parametr	Popis
<b>enabled</b>	Řízení spínače je povoleno v konfiguračním webovém rozhraní.
<b>mode</b>	Nastavený režim spínače ( <b>monostable</b> , <b>bistable</b> )
<b>switchOnDuration</b>	Doba sepnutí spínače v sekundách (jen pro monostabilní režim)
<b>type</b>	Typ spínače ( <b>normal</b> , <b>security</b> )

**Příklad:**

```
GET /api/switch/caps
{
  "success" : true,
  "result" : {
    "switches" : [
      {
        "switch" : 1,
        "enabled" : true,
        "mode" : "monostable",
        "switchOnDuration" : 5,
        "type" : "normal"
      },
      {
        "switch" : 2,
        "enabled" : true,
        "mode" : "monostable",
        "switchOnDuration" : 5,
        "type" : "normal"
      },
      {
        "switch" : 3,
        "enabled" : false
      },
      {
        "switch" : 4,
        "enabled" : false
      }
    ]
  }
}
```

## 5.4.2 api switch status

Funkce **api/switch/status** vrací aktuální stavy spínačů.

### Skupiny služeb a privilegií

- Skupina služeb je Switch.
- Skupina privilegií je Spínače – řízení.

### Metody

- GET
- POST

### Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>switch</b>	No	celé číslo definující spínač (obvykle 1 až 4)	–	Definuje spínač, jehož stav má být vrácen. Funkce <b>api/switch/caps</b> může sloužit ke zjištění počtu spínačů určitého zařízení. Jestliže je tento parametr vynechán, vrátí se stav všech spínačů.
<b>hold Timeout</b>	No		–	Definuje zbývající čas přidržení spínače. Parametr se v odpovědi nezobrazuje, pokud není nastaven časový limit nebo je spínač přidržen bez časového limitu.

### Příklad požadavku

URL: <https://192.168.1.1/api/switch/status?switch=1>

**Odpověď**

Úspěšná odpověď je ve formátu **application/json**. Obsahuje dva JSON klíče **success** a **result**, který obsahuje klíč **switches** (stavové informace jednotlivých spínačů jsou v poli s jedním až čtyřmi prvky).

Tabulka 2. Klíče JSON odpovědi switches

Klíč	Typické vrácené hodnoty	Popis
<b>switch</b>	celé číslo (obvykle 1 až 4)	Definuje, ke kterému spínači se stav vztahuje.
<b>active</b>	true nebo false	Definuje aktuální stav spínače (true – spínač je sepnutý, false – spínač je vypnutý).
<b>locked</b>	true nebo false	Definuje, zda je spínač uzamčený, nebo ne (true – spínač je uzamčený ve vypnutém stavu a nedá se ovládat, false – spínač je odemčený a dá se normálně ovládat). Uzamčení má přednost před přidržením spínače – tj. když je spínač současně uzamčený a přidržený, je vypnutý a nedá se ovládat.
<b>held</b>	true nebo false	Definuje, zda je spínač přidržený, nebo ne (true – spínač je přidržen v sepnutém stavu a nedá se ovládat, false – spínač je uvolněn a dá se normálně ovládat). Uzamčení má přednost před přidržením spínače – tj. když je spínač současně uzamčený a přidržený, je vypnutý a nedá se ovládat.

**Příklad odpovědi**

```
GET /api/switch/status { "success" : true, "result" : { "switches" : [ { "switch" : 1,
  "active" : true "locked" : false "held" : true }, { "switch" : 2, "active" : false
  "locked" : false "held" : false }, { "switch" : 3, "active" : false "locked" : true
  "held" : false }, { "switch" : 4, "active" : false "locked" : true "held" : true } ]
} }
```

Mohou se vyskytnout různé chyby (např. chybějící povinný parametr). Chyby se vrací ve formátu .json s odpovídajícím kódem 200.

### 5.4.3 api switch ctrl

Funkce **/api/switch/ctrl** se používá k ovládání spínačů.

#### Skupiny služeb a privilegií

- Skupina služeb je Switch.
- Skupina privilegií je Spínače (řízení).

#### Metody

- GET
- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>switch</b>	Ano	Celé číslo definující spínač (obvykle 1 až 4)	–	Definuje, který spínač má být ovládán. Funkce <b>api/switch/caps</b> může sloužit ke zjištění počtu spínačů určitého zařízení.

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>action</b>	Ano	Řetězec definující příkaz	-	<p>Definuje, který příkaz bude pro spínač použit. Lze použít tyto příkazy:</p> <ul style="list-style-type: none"> <li>• on – sepnutí spínače</li> <li>• off – vypnutí spínače</li> <li>• trigger – sepnutí monostabilního spínače, překlopení stavu bistabilního spínače</li> <li>• lock – uzamčení spínače (uzamčený spínač je vypnutý a nedá se ovládat)</li> <li>• unlock – odemknutí spínače (povolení normálního provozu)</li> <li>• hold – přidržení sepnutého spínače (přidržený spínač je sepnutý, ale nedá se ovládat), jestliže je spínač uzamčen a zároveň přidržen, je vypnutý</li> <li>• release – uvolnění spínače z přidržení (povolení normálního provozu)</li> </ul>
<b>response</b>	Ne	Řetězec definující text, který má být vrácen místo standardní odpovědi JSON	-	Zařízení vrátí text uvedený v tomto parametru místo standardní odpovědi JSON.
<b>time out</b>	Ne (zadává se při použití parametru action=hold)	Rozsah v sekundách 1–86 400.	-	Definuje dobu v sekundách, po které se spínač po přijetí příkazu hold automaticky opět uvolní.



## Příklad požadavku

```
URL: https://192.168.1.1/api/switch/ctrl?switch=4&action=trigger&response=TEST
```

## Odpověď

Úspěšná odpověď je ve formátu **application/json** (pokud není v parametru `response` uveden jiný text odpovědi).

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>success</b>	true nebo false	Jestliže byl příkaz proveden správně, je vrácená hodnota true, a jestliže nebylo možno dosáhnout požadovaného stavu, je false (např. když je spínač uzamčen a požadovaným stavem byl sepnutý spínač).

## Příklad odpovědi

```
{ "success": true }
```

Když je vrácena hodnota false, odpověď obsahuje další informace. Chybový kód 14, "akce se nezdařila", znamená, že nebylo možno dosáhnout požadovaného výsledku (např. když je spínač uzamčen a byla požadována akce `action=on`). Příkaz ke změně typu operace (tj. přidržení, uzamčení) bude vždy úspěšný, protože typ operace se dá změnit kdykoli s výjimkou případu, kdy je spínač zakázán (zařízení v takovém případě vrátí na všechny příkazy chybu 14).

## 5.5 api io

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/io**.

- [5.5.1 api io caps](#)
- [5.5.2 api io status](#)
- [5.5.3 api io ctrl](#)

### 5.5.1 api io caps

Funkce **/api/io/caps** vrací seznam dostupných hardwarových vstupů a výstupů zařízení (portů). Funkce má volitelný parametr **port**, který určuje vstup/výstup, jehož vlastnosti se mají vrátit. Pokud parametr **port** není uveden, funkce vrací seznam všech vstupů a výstupů.

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>Port</b>	Volitelný parametr definující identifikátor vstupu nebo výstupu.

Odpověď je ve formátu **application/json** a obsahuje seznam portů (pole **ports**) a jejich aktuální nastavení. V případě použití parametru **port** obsahuje pole **ports** právě jednu položku.

Parametr	Popis
<b>port</b>	Identifikátor vstupu nebo výstupu
<b>type</b>	Typ ( <b>input</b> – pro digitální vstupy, <b>output</b> – pro digitální výstupy)

#### **Příklad:**

```
GET /api/io/caps
{
  "success" : true,
  "result" : {
    "ports" : [
      {
        "port" : "relay1",
        "type" : "output"
      },
      {
        "port" : "relay2",
        "type" : "output"
      }
    ]
  }
}
```

### 5.5.2 api io status

Funkce **/api/io/status** vrací aktuální stav logických vstupů a výstupů zařízení (portů). Funkce má volitelný parametr **port**, který určuje vstup/výstup, jehož stav se má vrátit. Pokud parametr **port** není uveden, funkce vrací stav všech vstupů a výstupů.

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>port</b>	Volitelný parametr definující identifikátor vstupu nebo výstupu. Identifikátory dostupných vstupů a výstupů lze získat pomocí funkce <b>/api/io/caps</b> .

Odpověď je ve formátu **application/json** a obsahuje seznam portů (pole **ports**) a jejich aktuální stav (položka **state**). V případě použití parametru **port** obsahuje pole **ports** právě jednu položku.

#### **Příklad:**

```
GET /api/io/status
{
  "success" : true,
  "result" : {
    "ports" : [
      {
        "port" : "relay1",
        "state" : 0
      },
      {
        "port" : "relay2",
        "state" : 0
      }
    ]
  }
}
```

### 5.5.3 api io ctrl

Funkce **/api/io/ctrl** řídí stav logického výstupu zařízení. Funkce má povinný parametr **port**, který určuje řízený výstup a povinný parametr **action** definující akci provedenou nad spínačem (sepnutí, vypnutí).

Funkce je součástí služby **I/O** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Vstupy a výstupy – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>port</b>	Povinný parametr definující identifikátor vstupu nebo výstupu. Identifikátory dostupných vstupů a výstupů lze získat pomocí funkce <b>/api/io/caps</b> .
<b>action</b>	Povinný parametr definující akci ( <b>on</b> – sepnutí výstupu, log.1, <b>off</b> – vypnutí výstupu, log.0).
<b>response</b>	Nepovinný parametr umožňující modifikovat odpověď interkomu tak, aby neobsahovala JSON zprávu, ale obyčejný text odpovídající tomuto parametru.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/io/ctrl?port=relay1&action=on
{
  "success" : true
}
```

V případě použití parametru **response** odpověď interkomu neobsahuje zprávu **json**, server vrací odpověď typu text/plain se zadaným textem (zadaný text může být prázdný).

**Příklad:**

```
GET /api/io/ctrl?port=relay1&action=on&response=text
text
```

```
GET /api/io/ctrl?port=relay1&action=on&response=
```

## 5.6 api phone

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/phone**.

- [5.6.1 api phone status](#)
- [5.6.2 api phone callog](#)
- [5.6.3 api phone config](#)

### 5.6.1 api phone status

Funkce **/api/phone/status** slouží k získání aktuálního stavu SIP účtů zařízení.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>account</b>	Volitelný parametr definující identifikátor SIP účtu (1 nebo 2). Pokud parametr není uveden, funkce vrací stav všech SIP účtů.

Odpověď je ve formátu **application/json** a obsahuje seznam SIP účtů zařízení (pole **accounts**) a jejich aktuální stav. V případě použití parametru **account** obsahuje pole **accounts** právě jednu položku.

Parametr	Popis
<b>account</b>	Jednoznačný identifikátor SIP účtu (1 nebo 2).
<b>enabled</b>	Signalizuje, zda je SIP účet povolen.
<b>sipNumber</b>	Telefonní číslo SIP účtu.
<b>registrationEnabled</b>	Signalizuje, zda má SIP účet povolenou registraci.
<b>registered</b>	Signalizuje, zda je účet úspěšně zaregistrován u SIP registraru.

**Příklad:**

```

GET /api/phone/status GET /api/phone/status
{
  "success" : true,
  "result" : {
    "accounts" : [
      {
        "account" : 1,
        "enabled" : true,
        "sipNumber" : "5207",
        "registrationEnabled" : true,
        "registered" : true,
        "registerTime" : 1663585547
      },
      {
        "account" : 2,
        "enabled" : false,
        "sipNumber" : "",
        "registrationEnabled" : true,
        "registered" : false
      }
    ]
  }
}

```

### 5.6.2 api phone callog

Funkce **/api/phone/callog** umožňuje stahovat nebo mazat všechny nebo vybrané záznamy hovorů.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování**.

Přehled záznamů zobrazuje informace:

- Typ volání
  - Příchozí volání (spojené nebo odmítnuté)
  - Zmeškané volání (příchozí nevyzvednuté)
  - Vyzvednuto jinde (příchozí vyzvednuté na jiném zařízení)
  - Odchozí volání (bez ohledu na výsledek)
  - Zvonkové tlačítko
- Typ kontaktu (nastavení ikony kontaktu)
- ID volaného/volajícího
- Datum a čas volání

#### Metoda GET nebo POST

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje aktuální stav zařízení:

Parametr	Popis
id	Jednoznačná identifikace záznamu.
callType	Specifikuje typ volání. <ul style="list-style-type: none"><li>• incoming</li><li>• outgoing</li><li>• missed</li><li>• voicemail</li><li>• completedElsewhere</li><li>• zvonkové tlačítko</li></ul>
devType	Interní identifikátor zařízení.
name	Specifikuje jméno uživatele z telefonního seznamu.
date	Datum záznamu volání.
duration	Definuje dobu volání v sekundách.

Záznamy jsou seřazené od nejnovějšího k nejstaršímu podle absolutního času vytvořeného záznamu.

#### **Upozornění**

- Pole je prázdné v případě, že nejsou k dispozici žádné záznamy.

**Příklad:**

```
{
  "success" : true,
  "result" : {
    "callLog" : [
      {
        "id" : ID,
        "callType" : "incoming",
        "devType" : "2-14-0-0",
        "name" : "Franta Vomáčka",
        "date" : "2027-11-06T12:23:52Z",
        "duration": 1514
      },
      {
        "id" : ID,
        "callType" : "incoming",
        "devType" : "4-13-1-2",
        "name" : "Pepa Vonášek",
        "date" : "2027-12-06T12:23:52Z",
        "duration": 15
      },
      ...
    ]
  }
}
```

**Zvonkové tlačítko**

```
{
  "success" : true,
  "result" : {
    "callLog" : [
      {
        "id" : ID,
        "callType" : "doorbell",
        "date" : "2027-11-06T12:23:52Z"
      },
      ...
    ]
  }
}
```

**Metoda DELETE**

Funkce je součástí služby **Phone/Call API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Phone/Call Access Control**.

Parametry požadavku:



Parametr	Popis
<b>id</b>	Jednoznačná identifikace záznamu, který bude smazán.

**Příklad:**

```
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "id",
    "description" : "record not found"
  }
}
```

### 5.6.3 api phone config

Funkce **/api/phone/config** slouží ke sledování a kontrolování nastavení SIP účtů.

U této funkce lze použít metodu **GET** pro stažení a **PUT** pro nahrání konfigurace.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování** pro metodu **GET** a **Telefon/hovory – řízení** pro metodu **PUT**.

#### Metoda GET

Parametry požadavku:

Parametr	Popis
<b>account</b>	Volitelný parametr definující identifikátor SIP účtu (1 nebo 2). Pokud parametr není uveden, funkce vrací stav všech SIP účtů.

Pro metodu **GET** je odpověď ve formátu **application/json** a obsahuje seznam SIP účtů zařízení (pole **accounts**) a jejich aktuální stav. V případě použití specifikování účtu parametrem **account** odpověď obsahuje jen informace o daném účtu.

#### Upozornění

- Zařízení z bezpečnostních důvodů nevrací heslo při použití metody **GET**.

**Příklad:**

```

GET /api/phone/config
{
  "success": true,
  "result": {
    "accounts": [
      {
        "account": 1,
        "enabled": false,
        "displayName": "",
        "sipNumber": "",
        "domain": "",
        "domainPort": "",
        "authId": "",
        "proxyAddress": "",
        "proxyPort": "",
        "registrationEnabled": false,
        "registrarAddress": "",
        "registrarPort": "",
        "answerMode": "1"
      },
      {
        "account": 2,
        "enabled": false,
        "displayName": "",
        "sipNumber": "",
        "domain": "",
        "domainPort": "",
        "authId": "",
        "proxyAddress": "",
        "proxyPort": "",
        "registrationEnabled": false,
        "registrarAddress": "",
        "registrarPort": "",
        "answerMode": "1"
      }
    ]
  }
}

```

**Metoda PUT**

Parametry požadavku:

Parametr	Popis
<b>blob-json</b>	Povinný parametr obsahující konfiguraci SIP účtů (ve formátu JSON).

Pro metodu **PUT** je povinný parametr **blob-json**, který může obsahovat všechny parametry z pole **accounts** ze souboru získaného metodou **GET**. Kromě povinného parametru **account** musí obsahovat minimálně ještě jeden další parametr. Ostatní parametry jsou volitelné. U každého účtu v uploadovaném JSON souboru je možné specifikovat parametr **password** a zadat heslo v otevřené podobě. Tento parametr není z bezpečnostních důvodů součástí odpovědi na metodu **GET**. Odpověď je ve formátu **application/json**. Pokud se při ověřování vyskytne chyba, celý proces skončí neúspěšně a žádný z parametrů nebude použit.

### Příklad:

```
PUT /api/phone/config
{
  "success": true,
}
```

Parametry z databáze odpovídají parametrům v JSON souboru následovně:

Parametr z databáze	JSON parametr	Doplňkové informace
<b>Phone.Sip</b>	<b>account</b>	Číslování začíná od 1, ne od 0.
<b>Phone.Sip.Enabled</b>	<b>enabled</b>	
<b>Phone.Sip.User.DisplayName</b>	<b>displayName</b>	
<b>Phone.Sip.UserId</b>	<b>sipNumber</b>	
<b>Phone.Sip.User.AuthId</b>	<b>authId</b>	Pokud zůstane parametr prázdný, bude parametr Phone.Sip.UserId použit místo něj.
<b>Phone.Sip.User.PasswordString</b>	<b>password</b>	V otevřené podobě – lze do zařízení nahrát pouze pomocí funkce PUT, ale nelze jej získat pomocí funkce GET.
<b>Phone.Sip.Client.Domain</b>	<b>domain</b>	

Parametr z databáze	JSON parametr	Doplňkové informace
<b>Phone.Sip.Client.Port</b>	<b>domainPort</b>	
<b>Phone.Sip.Proxy.Address</b>	<b>proxyAddress</b>	
<b>Phone.Sip.Proxy.Port</b>	<b>proxyPort</b>	
<b>Phone.Sip.Registrar.Enabled</b>	<b>registrationEnabled</b>	
<b>Phone.Sip.Registrar.Address</b>	<b>registrarAddress</b>	
<b>Phone.Sip.Registrar.Port</b>	<b>registrarPort</b>	
<b>Phone.Sip.Misc.AnswerMode</b>	<b>answerMode</b>	

## 5.7 api call

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/call**.

- [5.7.1 api call status](#)
- [5.7.2 api call dial](#)
- [5.7.3 api call answer](#)
- [5.7.4 api call hangup](#)

### 5.7.1 api call status

Funkce **/api/call/status** slouží k získání aktuálního stavu probíhajících telefonního hovorů. Funkce vrací seznam probíhajících hovorů a jejich parametry.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>session</b>	Volitelný parametr obsahující identifikátor hovoru, jehož stav se má vrátit. Pokud parametr není uveden, funkce vrací stav všech probíhajících hovorů.

Odpověď je ve formátu **application/json** a obsahuje seznam probíhajících hovorů (pole **sessions**) a jejich aktuální stav. V případě použití parametru **session** obsahuje pole **sessions** právě jednu položku. Pokud aktuálně neprobíhá žádný hovor, pole **sessions** je prázdné.

Parametr	Popis
<b>session</b>	Identifikátor hovoru.
<b>direction</b>	Směr hovoru ( <b>incoming</b> – příchozí, <b>outgoing</b> – odchozí)
<b>state</b>	Stav hovoru ( <b>connecting</b> , <b>ringing</b> , <b>connected</b> )

#### Příklad:

```
GET /api/call/status
{
  "success" : true,
  "result" : {
    "sessions" : [
      {
        "session" : 1,
        "direction" : "outgoing",
        "state" : "ringing"
      }
    ]
  }
}
```

### 5.7.2 api call dial

Funkce **/api/call/dial** umožňuje iniciovat nový odchozí hovor na zvolené telefonní číslo nebo sip uri pomocí parametru *number* nebo na jednoho či více uživatelů pomocí parametru *users*. Příkaz smí obsahovat pouze jeden z uvedených parametrů, v opačném případě bude vrácena odpověď s chybovým hlášením.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>number</b>	Povinný parametr specifikující cílové telefonní číslo nebo sip uri.
<b>users</b>	Seznam uuid (unikátní ID) uživatelů oddělenými čárkami.

Odpověď je ve formátu **application/json** a obsahuje informace o vytvořeném odchozím hovoru.

Parametr	Popis
<b>session</b>	Identifikátor hovoru, který lze použít např. pro sledování hovoru pomocí funkce <b>/api/call/status</b> , příp. pro ukončení hovoru funkcí <b>/api/call/hangup</b> .

#### **Příklad:**

```
GET /api/call/dial?number=sip:1234@10.0.23.194
{
  "success" : true,
  "result" : {
    "session" : 2
  }
}
```

### 5.7.3 api call answer

Funkce **/api/call/answer** umožňuje vyzvednout probíhající příchozí hovor (ve stavu **ringing**).

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>session</b>	Identifikátor probíhajícího příchozího hovoru.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/call/answer?session=3
{
  "success" : true
}
```

### 5.7.4 api call hangup

Funkce **/api/call/hangup** umožňuje ukončit probíhající příchozí nebo odchozí hovor.

Funkce je součástí služby **Phone/Call** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Telefon/hovory – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>session</b>	Identifikátor probíhajícího příchozího nebo odchozího hovoru.
<b>reason</b>	Důvod ukončení hovoru: <b>"normal"</b> – běžné ukončení hovoru (výchozí hodnota) <b>"rejected"</b> – signalizace odmítnutí hovoru <b>"busy"</b> – signalizace obsazení stanice

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/call/hangup?session=4
{
  "success" : true
}
```

### 5.8 api camera

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/camera**.

- [5.8.1 api camera caps](#)
- [5.8.2 api camera snapshot](#)

### 5.8.1 api camera caps

Funkce **/api/camera/caps** vrací seznam možných zdrojů videa a variant rozlišení JPEG snímků, které lze stahovat pomocí funkce **/api/camera/snapshot**.

Funkce je součástí služby **Camera** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Kamera – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje seznam podporovaných rozlišení JPEG snímků (pole **jpegResolution**) a seznam dostupných zdrojů obrazu (pole **sources**), které lze použít v parametrech funkce **/api/camera/snapshot**.

Parametr	Popis
<b>width, height</b>	Rozlišení snímku v pixelech
<b>source</b>	Identifikátor zdroje obrazu



**Příklad:**

```
GET /api/camera/caps
{
  "success" : true,
  "result" : {
    "jpegResolution" : [
      {
        "width" : 160,
        "height" : 120
      },
      {
        "width" : 176,
        "height" : 144
      },
      {
        "width" : 320,
        "height" : 240
      },
      {
        "width" : 352,
        "height" : 272
      },
      {
        "width" : 352,
        "height" : 288
      },
      {
        "width" : 640,
        "height" : 480
      }
    ],
    "sources" : [
      {
        "source" : "internal"
      },
      {
        "source" : "external"
      }
    ]
  }
}
```

### 5.8.2 api camera snapshot

Funkce **/api/camera/snapshot** umožňuje stažení obrázku z interní nebo externí IP kamery připojené k interkomu. Pomocí parametrů lze specifikovat zdroj obrázku, rozlišení apod.

Funkce je součástí služby **Camera** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Kamera – sledování**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parameter	Popis
<b>width</b>	Povinný parametr specifikující horizontální rozlišení JPEG snímku v pixelech. Rozlišení musí odpovídat jedné z podporovaných variant (viz funkce <b>api/camera/caps</b> ). Při zadání nepodporované hodnoty nebude požadavek proveden.
<b>height</b>	Povinný parametr specifikující vertikální rozlišení JPEG snímku v pixelech. Rozlišení snímku musí odpovídat jedné z podporovaných variant (viz funkce <b>api/camera/caps</b> ). Při zadání nepodporované hodnoty nebude požadavek proveden.
<b>source</b>	Volitelný parametr definující zdroj videa ( <b>internal</b> – interní kamera, <b>external</b> – externí IP kamera). Pokud parametr není uveden, je zvolen výchozí zdroj videa uvedený v konfiguračním webovém rozhraní v sekci Hardware / Kamera / Společné nastavení.
<b>fps</b>	Volitelný parametr definující snímkovou frekvenci. Pokud je parametr nastaven na hodnotu $\geq 1$ , interkom odesílá s nastavenou snímkovou frekvencí obrázky metodou <b>http server push</b> .
<b>time</b>	Volitelný parametr definující čas snímku v paměti vrátníku. Hodnoty time musí být v rozsahu paměti vrátníku, tedy $\langle -30, 0 \rangle$ sekund. Pokud je tento parametr použit společně s parametrem <b>fps</b> , je parametr <b>fps</b> ignorován a funkce vrátí pouze jeden snímek.  Pokud je tento parametr použit, vrací funkce snímky o maximálním rozlišení 1280 x 960 px (width x height). Požadavky parametrů <b>width</b> a <b>height</b> na vyšší hodnoty budou ignorovány.

Odpověď je ve formátu **image/jpeg** příp. **multipart/x-mixed-replace** (pro  $\text{fps} \geq 1$ ). V případě chybných parametrů požadavku, funkce vrací informaci ve formátu **application/json**.

**Příklad:**

```
GET /api/camera/snapshot?width=640&height=480&source=internal
```

# následující příkaz vrátí snímek zachycený 5 sekund před voláním funkce

```
GET /api/camera/snapshot?width=640&height=480&source=internal&time=-5
```

## 5.9 api display

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/display**.

- [5.9.1 api display caps](#)
- [5.9.2 api display image](#)

### 5.9.1 api display caps

Funkce **/api/display/caps** vrací seznam displejů v zařízení a jejich vlastnosti. Funkci lze použít pro detekci displeje a získání jeho rozlišení.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej – řízení**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a obsahuje seznam dostupný displejů (pole **displays**).

Parametr	Popis
<b>display</b>	Identifikátor displeje
<b>resolution</b>	Rozlišení displeje v pixelech

**Příklad:**

```
GET /api/display/caps
{
  "success" : true,
  "result" : {
    "displays" : [
      {
        "display" : "internal",
        "resolution" : {
          "width" : 320,
          "height" : 240
        }
      }
    ]
  }
}
```

## 5.9.2 api display image

2N<sup>®</sup> IP Style

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát obrázek, příp. nahraný obrázek z displeje odstranit. JPEG obrázky s progresivní kvalitou nejsou podporovány.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.

**Metoda PUT****Parametry požadavku:**

Parametr	Popis
<b>blob-image</b>	Povinný parametr obsahující obrázek ve formátu JPEG, PNG s rozlišením daného displeje 1280 x 800 pixelů (viz funkce <b>/api/display/caps</b> ). Parametr se uplatní pouze v případě metody <b>PUT</b> . Progresivní JPEG obrázky nejsou podporovány.
<b>duration</b>	Volitelný parametr. Doba zobrazení obrázku. Parametr se nastavuje v milisekundách.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazena jiným obrázkem nebo uživatel sám obrázek odstraní.

Pokud HTTP požadavek neobsahuje volitelný parametr, jedná se o zobrazení v režimu overlay, tj. obrázek nahraný na neurčitou dobu. Pokud je volitelný parametr uveden, obrázek se zobrazí jako notifikace, která se ukončí po uplynutí nastavené doby. Zobrazení notifikace je možné předčasně ukončit dotykem displeje.

Nahrává-li se obrázek poprvé, přenáší se z hlavní jednotky na displej přes interní sběrnici (což může chvíli trvat). V paměti displeje se dá uložit i několik obrázků, a pokud se někdy takto uložené obrázky budou znovu posílat do zařízení, nebude je již třeba přenášet přes interní sběrnici. Místo toho je bude možno ihned zobrazit na displeji z paměti.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

### **Parametry obrázku:**

Model	Rozměry obrázku	Podporované formáty
2N <sup>®</sup> IP Style	1280 x 800 pixelů	normální JPEG (doporučeno), PNG

### Upozornění

- Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

### **Příklad:**

```
PUT api/display/image&duration=30000
{
  "success" : true
}
```

**Metoda DELETE**

Parametr	Popis
<b>display</b>	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze zjistit hodnotu pomocí funkce <b>/api/display/caps</b> .

**Příklad:**

```
DELETE /api/display/image
{
  "success" : true
}
```

2N<sup>®</sup> IP Verso

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát, příp. nahraný obrázek z displeje odstranit. Progresivní JPEG obrázky nejsou podporovány.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.

**Metoda PUT****Parametry požadavku:**

Parametr	Popis
<b>display</b>	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze hodnotu zjistit pomocí funkce <b>/api/display/caps</b> .
<b>blob-image</b>	Povinný parametr obsahující obrázek ve formátu JPEG, BMP, PNG s rozlišením daného displeje 320 x 214 pixelů (viz funkce <b>/api/display/caps</b> ). Parametr se uplatní pouze v případě metody <b>PUT</b> . Požadavek smí obsahovat pouze jeden z parametrů, blob-image nebo blob-video. Progresivní JPEG obrázky nejsou podporovány.

Parametr	Popis
<b>blob-video</b>	Povinný parametr obsahující video ve formátu MPEG4 / H264, max. délka 60 s, fps max. 15 nahraný v rozlišení 320 x 214 pixelů. Požadavek smí obsahovat pouze jeden z parametrů, blob-image nebo blob-video.
<b>duration</b>	Volitelný parametr. Doba zobrazení obrázku/přehrávání videa. Parametr se nastavuje v milisekundách.
<b>repeat</b>	Volitelný parametr. Počet opakování přehrávání videa. Parametr se týká pouze videa.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazena jiným obrázkem nebo uživatel sám obrázek odstraní.

Pokud HTTP požadavek neobsahuje ani jeden z výše uvedených parametrů, jedná se o zobrazení v režimu overlay tj. obrázek nahraný na neurčitou dobu. Pokud jsou uvedeny oba volitelné parametry, zobrazení notifikace se ukončí událostí, která nastane dříve. Zobrazení notifikace je možné předčasně ukončit dotykem displeje.

Nahrává-li se obrázek poprvé, přenáší se z hlavní jednotky na displej přes interní sběrnici (což může chvíli trvat). V paměti displeje se dá uložit i několik obrázků, a pokud se někdy takto uložené obrázky budou znovu posílat do zařízení, nebude je již třeba přenášet přes interní sběrnici. Místo toho je bude možno ihned zobrazit na displeji z paměti.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

#### **Parametry obrázku:**

Model	Rozměry obrázku	Podporované formáty
<b>2N<sup>®</sup> IP Verso</b>	320 x 214 pixelů	normální JPEG (doporučeno), BMP, PNG

#### Upozornění

- Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

**Příklad:**

```

api/display/image?display=ext1&duration=30000
{
  "success" : true
}

```

**Parametry videa:**

Model	Rozměry videa	Podporované formáty
2N <sup>®</sup> IP Verso	320 x 214 pixelů	MPEG4 / H264: Baseline profil, level max. 5.2

**Metoda DELETE**

Paramet r	Popis
<b>display</b>	Povinný identifikátor displeje. Hodnota je nastavena v parametru Hardware / Rozšiřující moduly / Jméno modulu. Případně lze zjistit hodnotu pomocí funkce / <b>api/display/caps</b> .

**Příklad:**

```

DELETE /api/display/image
{
  "success" : true
}

```

2N<sup>®</sup> IP Vario

Funkce **/api/display/image** umožňuje modifikovat obsah zobrazovaný na displeji zařízení. Umožňuje nahrát, příp. nahraný obrázek z displeje odstranit.

Funkce je součástí služby **Displej** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Displej (řízení)**.

Pro tuto funkci lze použít metody **PUT** nebo **DELETE**. Metoda **PUT** slouží k uploadu obrázku na displej. Metoda **DELETE** slouží k odstranění dříve uploadovaného obrázku z displeje.



**Parametry požadavku:**

Parametr	Popis
<b>display</b>	Povinný identifikátor displeje ( <b>internal</b> )
<b>blob-image</b>	Povinný parametr obsahující obrázek v podporovaném formátu s rozlišením daného displeje (viz funkce <b>/api/display/caps</b> ). Parametr se uplatní pouze v případě metody <b>PUT</b> .

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Parametry obrázku:**

Model	Rozměry obrázku	Podporované formáty
<b>2N® IP Vario</b>	320 x 240 pixelů	JPEG (doporučeno), GIF, BMP

## Upozornění

Podporovaný JPEG formát je JPEG Baseline (neprogresivní kódování).

## 5.9.2.1 Příklady api display image

Uvedené příklady slouží jako vzor dat zasílaných z řídicí aplikace na displej **2N® IP Verso** a **2N® IP Vario**.

Obrázek lze zobrazit dvěma způsoby jako notifikace nebo overlay. Oba způsoby jsou platné pouze pro model **2N® IP Verso**, u modelu **2N® IP Vario** lze obrázek zobrazit pouze přes notifikaci. Notifikace se zobrazuje na předem definovanou dobu, po které automaticky zmizí. Overlay zůstává na displeji do té doby, než je nahrazen jiným obrázkem, nebo uživatel sám obrázek odstraní.

Parameter **\*duration\*** udává dobu zobrazení obrázku/video v ms.

Parametr **\*repeat\*** udává počet opakování videa a u obrázku se ignoruje.

Pokud HTTP požadavek neobsahuje ani jeden z výše uvedených parametrů, jedná se o zobrazení v režimu overlay. Tj. obrázek nahraný na neurčitou dobu. Pokud jsou uvedeny oba parametry, zobrazení se ukončí událostí, která nastane dříve. e

Nahrání obrázku na displej 2N<sup>®</sup> IP Verso a 2N<sup>®</sup> IP Vario**i Poznámka**

Každý z modelů podporuje jiné rozlišení obrázků.

Model	Rozměry obrázku	Podporované formáty
<b>2N<sup>®</sup> IP Verso</b>	214 x 240 pixelů	JPEG (doporučeno), BMP, PNG
<b>2N<sup>®</sup> IP Vario</b>	320 x 240 pixelů	JPEG (doporučeno), GIF, BMP

URL požadavku: <https://10.27.24.15/api/display/image?display=ext1>

- Metoda požadavku: PUT
- Vzdálená adresa: 10.27.24.15:443
- Stavový kód: 200 OK
- Verze: HTTP/1.1

Hlavičky odpovědi (95 B)

- Server: HIP2.22.0.31.1
- Content-Type: application/json
- Content-Length: 24

Hlavičky požadavku (494 B)

- Host: 10.27.24.15
- User-Agent: Mozilla/5.0 (Windows NT 6.1; W...) Gecko/20100101 Firefox/56.0
- Accept: \*/\*
- Accept-Language: cs,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate, br
- Referer: <https://10.27.24.15/apitest.html>
- Content-Length: 1325
- Content-Type: multipart/form-data; boundary=...-----258852674219952
- Cookie: \_ga=GA1.1.375392382.1496656977...id=GA1.1.638680516.1507547865
- Connection: keep-alive



Nahrání videa na displej 2N<sup>®</sup> IP Verso**i** Poznámka

Model	Rozměry videa	Podporované formáty
<b>2N<sup>®</sup> IP Verso</b>	214 x 240 pixelů	MPEG4 / H264: Baseline profil, level max. 5.2

URL požadavku: <https://10.27.24.15/api/display/image?display=ext1&duration=20&repeat=3>

- Metoda požadavku: PUT
- Vzdálená adresa: 10.27.24.15:443
- Stavový kód: 200 OK
- Verze: HTTP/1.1

Hlavičky odpovědi (95 B)

- Server: HIP2.22.0.31.1
- Content-Type: application/json
- Content-Length: 24

Hlavičky požadavku (516 B)

- Host: 10.27.24.15
- User-Agent: Mozilla/5.0 (Windows NT 6.1; W...) Gecko/20100101 Firefox/56.0
- Accept: \*/\*
- Accept-Language: cs,en-US;q=0.7,en;q=0.3
- Accept-Encoding: gzip, deflate, br
- Referer: <https://10.27.24.15/apitest.html>
- Content-Length: 943815
- Content-Type: multipart/form-data; boundary=-----14948718218673
- Cookie: \_ga=GA1.1.375392382.1496656977...id=GA1.1.638680516.1507547865
- Connection: keep-alive



## 5.10.1 api log caps

Funkce **/api/log/caps** vrací seznam typů podporovaných událostí, které se na daném zařízení zaznamenávají. Vrácený seznam je v závislosti na zařízení podmnožinou kompletního seznamu typů událostí uvedeného v následující tabulce:

Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>AccessBlocked</b>	Signalizuje blokování autentizace uživatele, zónového kódu, odchodového tlačítka REX a přístupu pomocí registrační značky vozidla na přístupovém bodu.	"ap, state"	
<b>AccessLimited</b>	Událost, která nastane po zadání 5 neúspěšných pokusů o autentizaci uživatele (karta, kód, otisk prstu). Přístupový modul bude zablokován po dobu 30 sekund i v případě, že následná autentizace by byla správná. Signalizuje odmítnutí zadaného uživatele.	"ap, type, state"	
<b>AccessTaken</b>	Při přiložení karty v Anti-passback oblasti.	"ap, session"	
<b>ApiAccessRequested</b>	Událost, kdy byl zaslán požadavek na /api/accesspoint/grantaccess s výsledkem "success" : true.	"ap, valid"	"session, uuid"
<b>AudioLoopTest</b>	Signalizuje provedení automatického audio loop testu a jeho výsledek.	"result"	
<b>CallSessionStateChanged</b>	Událost popisující směr, stav hovoru, adresu, číslo vytvořené session a kolikátý hovor se generoval.	"session, state"	"originator, info"
<b>CallStateChanged</b>	Při změně stavu hovoru (ringing, connected, terminated) indikuje i směr (příchozí, odchozí) a identifikaci protistrany nebo účtu SIP.	"direction, state, peer, session, call"	"reason, device, sipAccount, sipCallId "
<b>CapabilitiesChanged</b>	Signalizuje změnu dostupných funkcí.		

Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>CardHeld</b>	Signalizuje přiložení RFID karty ke čtečce delší než 4 s.	"reader, uid, valid"	"ap, session, direction, uuid"
<b>CardEntered</b>	Signalizuje přiložení RFID karty ke čtečce.	"reader, uid, valid"	"ap, session, direction, uuid"
<b>CodeEntered</b>	Signalizuje zadání kódu uživatelem pomocí numerické klávesnice.	"code, valid "	"ap, session, direction, input, type, uuid, reason"
<b>ConfigurationChanged</b>	Změna nastavení konfigurace zařízení.		
<b>DeviceState</b>	Systémová událost generovaná při změnách stavu zařízení.	"state"	
<b>DisplayTouched</b>	Signalizuje dotyk na displeji.	"x, y, dx, dy"	
<b>DirectoryChanged</b>	Změna v adresáři.	"series"	"timestamp"
<b>DirectorySaved</b>	Uložení změny v adresáři.	"series"	"timestamp"
<b>DoorOpenTooLong</b>	Signalizuje dlouhé otevření dveří, resp. nezavření dveří do nastavené doby.	"state"	
<b>DoorStateChanged</b>	Signalizuje změnu stavu dveří.	"state"	
<b>DtmfEntered</b>	Příjem DTMF kódu v hovoru nebo lokálně mimo hovor.	"code, call, valid"	"type, uuid"

Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>DtmfPressed</b>	Zadání DTMF kódu v hovoru nebo lokálně mimo hovor.	"code, call valid"	
<b>DtmfSent</b>	Odeslání FTMF kódu v hovoru nebo lokálně mimo hovor.	"code"	"call"
<b>ExternalCameraStateChanged</b>	Signalizuje změnu stavu připojené externí kamery.	"state"	"id, reason"
<b>ErrorStateChanged</b>	Signalizuje změnu chybného stavu LiftIP 2.0.		"in, state, reason"
<b>FingerEntered</b>	Signalizuje přiložení prstu k biometrické čtečce.	"valid"	"ap, session, direction, uuid"
<b>FingerEnrollState</b>	Přiložení prstu na čtečku pro nahrání otisku uživatele.	"session, state"	
<b>HardwareChanged</b>	Změna připojení rozšiřujících modulů.	"reason, class, id"	"info, config, state, categories"
<b>CheckingCall</b>	Zobrazuje detaily uskutečněného kontrolního volání.		"action"
<b>InputChanged</b>	Signalizuje změnu stavu logického vstupu.	"port, state"	
<b>KeyPressed</b>	Signalizuje stisk tlačítka rychlé volby, klávesy numerické klávesnice, dotyk na displeji nebo stisk tlačítka zahajujícího Bluetooth autentizaci.	"key"	
<b>KeyReleased</b>	Signalizuje uvolnění tlačítka rychlé volby nebo klávesy numerické klávesnice.	"key"	



Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>LicensePlate Recognized</b>	Signalizuje rozpoznání registrační značky vozidla s platnými právy pro přístup.	"ap, licensePlate, valid"	"session, uuid"
<b>LiftConfigChanged</b>	Změna nastavení řízení výtahu.	"hash"	
<b>LiftErrorStateChanged</b>			"in, state, reason"
<b>LiftFloorsEnabled</b>	Přístup na patro pomocí výtahu.	"floors"	"uuid, session"
<b>LiftStatusChanged</b>	Detekce připojení/odpojení Lift Control modulu.	"module, ready"	
<b>LoginBlocked</b>	Signalizuje dočasné zablokování přístupu k webovému rozhraní.	"address"	
<b>MobKeyEntered</b>	Signalizuje autorizaci pomocí bluetooth čtečky.	"action, authid, valid"	"ap, session, direction, uuid"
<b>MotionDetected</b>	Signalizuje detekci pohybu pomocí kamery.	ID = odpovídá číslu profilu detekce pohybu ve webovém rozhraní "state"	
<b>NoiseDetected</b>	Signalizuje detekci zvýšené hladiny hluku	pouze modely vybavené mikrofonem nebo mikrofonním vstupem "state"	

Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>OutputChanged</b>	Signalizuje změnu stavu logického výstupu.	"port, state"	
<b>PairingState Changed</b>	Signalizuje párování s bluetooth rozhraním.	"state, authId"	
<b>RescueState Changed</b>	Signalizuje změnu stavu režimu vyproštění.		"state, reason"
<b>Registration StateChange d</b>	Změna stavu registrace k SIP proxy.	"sipAccount, state"	"reason"
<b>RexActivated</b>	Signalizuje aktivaci odchodového tlačítka REX.	"ap, session, valid"	"reason"
<b>SilentAlarm</b>	Signalizuje aktivaci tichého alarmu.	"ap, session, name"	"uuid"
<b>SwitchesBlocked</b>	Signalizuje zablokování zámků tamper spínačem.	"state"	
<b>SwitchOperationChanged</b>	Změna fungování spínače (signalizuje stav uzamčení nebo přidržení spínače, nastartování i restartování časovače nebo jeho ukončení – přechodu do trvalého přidržení).	"switch"	"enabled, locked, held, hold_timeout, originator"
<b>SwitchState Changed</b>	Signalizuje změnu stavu spínače 1–4	"switch, state"	"ap, session, originator, call, peer, device"
<b>TamperSwitchActivated</b>	Signalizuje aktivaci ochranného spínače.	"state"	

Typ události	Popis	Parametry	
		Stálé	Podmíněné
<b>UnauthorizedDoorOpen</b>	Signalizuje neautorizované otevření dveří.	pouze modely vybavené digitálními vstupy "state"	
<b>UserActionActivated</b>	Signalizuje změnu stavu vstupu, který je nakonfigurován na funkci Spouštěče uživatelské akce.	"id, state"	
<b>UserAuthenticated</b>	Signalizuje autentizaci uživatele a následné otevření dveří.	"ap, session, name"	"uuid, apbBroken"
<b>UserRejected</b>	Signalizuje odmítnutí autorizace uživatele.	"ap, session, name"	"uuid, reason"
<b>VirtualInput</b>	Změna virtuálního vstupu.	"port, state"	
<b>VirtualOutput</b>	Změna virtuálního výstupu.	"port, state"	
<b>WaveKeyActivated</b>	Aktivace Bluetooth autentizace.	"type"	

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json**:

Parametr	Typ	Popis
<b>events</b>	array	Pole řetězců obsahující seznam podporovaných typů událostí.

**Příklad:**

```

GET /api/log/caps
{
  "success" : true,
  "result" : {
    "events" : [
      "KeyPressed",
      "KeyReleased",
      "InputChanged",
      "OutputChanged",
      "CardEntered",
      "CallStateChanged",
      "AudioLoopTest",
      "CodeEntered",
      "DeviceState",
      "RegistrationStateChanged"
    ]
  }
}

```

**5.10.2 api log subscribe**

Funkce **/api/log/subscribe** vytvoří kanál pro odběr událostí (subscription) a vrací unikátní identifikátor, který se použije při následném volání funkcí **/api/log/pull**, příp. **/api/log/unsubscribe**.

Každý kanál pro odběr událostí obsahuje vlastní frontu událostí. Do fronty kanálu jsou ukládány všechny nové události, které odpovídají filtru kanálu (parametr **filter**). Události z fronty kanálu lze číst pomocí funkce **/api/log/pull**.

Současně zařízení udržují v interní paměti frontu historie událostí (posledních 10000 událostí). Po restartu zařízení je tato fronta historie vždy prázdná.

Pomocí parametru **include** lze specifikovat, zda fronta kanálu bude na počátku prázdná (tj. budou do ní zapsány pouze nové události, které vzniknou po vytvoření kanálu), příp. zda má být jednorázově naplněna událostmi z části nebo celé zaznamenané historie událostí.

Pomocí parametru **duration** lze specifikovat životnost kanálu v případě, že se k němu nepřístupuje pomocí funkce **/api/log/pull**. Po nastavené době bude nepoužívaný kanál automaticky uzavřen, jako by byla použita funkce **/api/log/unsubscribe**.

Funkce je součástí služby **Logging** a v případě použití autentizace je nutné pro některé události nastavit privilegia uživatele podle tabulky níže. Do fronty kanálu nebudou zařazovány události, pro které autentizovaný uživatel nemá požadovaná privilegia.

**Tabulka událostí:**

Dostupné události	Vyžadovaná privilegia uživatele
<b>KeyPressed</b>	Sledování – Klávesnice
<b>KeyReleased</b>	Sledování – Klávesnice
<b>CodeEntered</b>	Sledování – Klávesnice
<b>TamperSwitchActivated</b>	žádná
<b>UnauthorizedDoorOpen</b>	žádná
<b>DoorOpenTooLong</b>	žádná
<b>LoginBlocked</b>	žádná
<b>SilentAlarm</b>	žádná
<b>DoorStateChanged</b>	žádná
<b>DeviceState</b>	žádná
<b>AudioLoopTest</b>	žádná
<b>MotionDetected</b>	žádná
<b>NoiseDetected</b>	žádná
<b>HardwareChanged</b>	žádná
<b>FingerEnrollState</b>	žádná
<b>LiftStatusChanged</b>	žádná
<b>LiftFloorsEnabled</b>	žádná
<b>LiftConfigChanged</b>	žádná
<b>CapabilitiesChanged</b>	žádná
<b>ConfigurationChanged</b>	žádná
<b>ExtCameraStateChanged</b>	žádná
<b>RescueStateChanged</b>	žádná

Dostupné události	Vyžadovaná privilegia uživatele
<b>ErrorStateChanged</b>	Žádná
<b>LiftCheckingCall</b>	Žádná
<b>DtmfSent</b>	Žádná
<b>RexActivated</b>	Žádná
<b>AccessBlocked</b>	Žádná
<b>AccessTaken</b>	Žádná
<b>AccessLimited</b>	Žádná
<b>DisplayTouched</b>	Žádná
<b>DtmfPressed</b>	Žádná
<b>SwitchesBlocked</b>	Žádná
<b>CardEntered</b>	Sledování – UID (karty/wiegand)
<b>CardHeld</b>	Sledování – UID (karty/wiegand)
<b>DtmfEntered</b>	Sledování – UID (karty/wiegand)
<b>PairingStateChanged</b>	Sledování – UID (karty/wiegand)
<b>MobKeyEntered</b>	Sledování – UID (karty/wiegand)
<b>WaveKeyEntered</b>	Sledování – UID (karty/wiegand)
<b>FingerEntered</b>	Sledování – UID (karty/wiegand)
<b>UserAuthenticated</b>	Sledování – UID (karty/wiegand)
<b>UserRejected</b>	Sledování – UID (karty/wiegand)
<b>ApiAccessRequested</b>	Sledování – Správa přístupu
<b>LicensePlateRecognized</b>	Sledování – Správa přístupu

Dostupné události	Vyžadovaná privilegia uživatele
<b>CallStateChanged</b>	Sledování – Telefon/Hovory
<b>CallSessionStateChanged</b>	Sledování – Telefon/Hovory
<b>RegistrationStateChanged</b>	Sledování – Telefon/Hovory
<b>InputChanged</b>	Sledování – Vstupy a výstupy
<b>OutputChanged</b>	Sledování – Vstupy a výstupy
<b>VirtualInput</b>	Sledování – vstupy a výstupy
<b>VirtualOutput</b>	Sledování – Vstupy a výstupy
<b>SwitchStateChanged</b>	Sledování – vstupy a výstupy
<b>SwitchOperationChanged</b>	Sledování – vstupy a výstupy
<b>UserActionActivated</b>	Sledování – Vstupy a výstupy
<b>DirectoryChanged</b>	Sledování – Systém
<b>DirectorySaved</b>	Sledování – Systém

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
<b>include</b>	string	Ne	new	<p>Určuje, zda má být fronta událostí kanálu naplněna položkami z historie:</p> <p><b>new</b> – pouze nové události, které vzniknou až po vytvoření kanálu</p> <p><b>all</b> – všechny dosud zaznamenané události včetně těch, které vzniknou až po vytvoření kanálu</p> <p><b>-t</b> – všechny dosud zaznamenané události za posledních <b>t</b> sekund včetně těch, které vzniknou až po vytvoření kanálu (např. -10)</p>
<b>filter</b>	list	Ne	bez filtru	<p>Seznam typů požadovaných událostí (názvy typů událostí oddělené čárkou). Parametr je nepovinný, a pokud není uveden, pak se v rámci kanálu předávají všechny typy událostí daného zařízení, které nejsou defaultně skryté. Pro odběr defaultně skrytých událostí je nutné události v tomto parametru vyžádat.</p> <p>Defaultně skryté události jsou:</p> <ul style="list-style-type: none"> <li>• <b>FingerEnrollState</b></li> <li>• <b>DirectorySaved</b></li> <li>• <b>DirectoryChanged</b></li> <li>• <b>HardwareChanged</b></li> <li>• <b>DisplayTouched</b></li> <li>• <b>PairingStateChanged</b></li> <li>• <b>LiftConfigChanged</b></li> <li>• <b>CapabilitiesChanged</b></li> <li>• <b>ConfigurationChanged</b></li> <li>• <b>ExtCameraStateChanged</b></li> </ul>



Parametr	Typ	Povinný	Výchozí hodnota	Popis
<b>duration</b>	uint32	Ne	90	Definuje dobu v sekundách, po které bude kanál automaticky uzavřen, pokud na něm nebudou probíhat žádné operace čtení pomocí <b>/api/log/pull</b> . Každým čtením z kanálu je automaticky životnost kanálu prodloužena o zde nastavenou hodnotu. Maximální možná hodnota je 3600 s.

Odpověď je ve formátu **application/json** a obsahuje pouze identifikátor vytvořeného subscription.

Parametr	Typ	Popis
<b>id</b>	uint32	Unikátní identifikátor vytvořeného subscription.

#### **Příklad:**

```
GET /api/log/subscribe?filter=KeyPressed,InputChanged
{
  "success" : true,
  "result" : {
    "id" : 2121013117
  }
}
```

### 5.10.3 api log unsubscribe

Funkce **/api/log/unsubscribe** uzavře kanál odběru událostí (subscription) s daným identifikátorem. Po provedení funkce nebude možné daný identifikátor použít, tj. následná volání funkce **/api/log/pull**, příp. **/api/log/unsubscribe** se stejným identifikátorem skončí chybou.

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
<b>id</b>	uint32	Ano	–	Identifikátor existujícího kanálu získaný při předchozím volání funkce <b>/api/log/subscribe</b> .

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/log/unsubscribe?id=21458715
{
  "success" : true,
}
```

#### 5.10.4 api log pull

Funkce **/api/log/pull** provádí čtení položek z fronty kanálu (subscription) a vrací seznam dosud nevyčtených událostí, příp. prázdný seznam, pokud žádná nová událost není k dispozici. Větší množství událostí jsou pak stahovány po dávkách o 128 událostech.

Pomocí parametru **timeout** lze specifikovat maximální dobu, za jakou musí interkom vygenerovat odpověď. V případě, že ve frontě kanálu je alespoň jedna položka, odpověď je vygenerována okamžitě. V případě, že je fronta kanálu prázdná, interkom odloží odeslání odpovědi do doby, než vznikne nová událost, příp. vyprší nastavený timeout.

Funkce je součástí služby **Logging** a pro provedení funkce nejsou potřeba žádná zvláštní privilegia uživatele. Čtení událostí je podmíněno privilegií uživatele tyto události sledovat, viz tabulka událostí v [5.10.2 api log subscribe](#).

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Typ	Povinný	Výchozí hodnota	Popis
<b>id</b>	uint32	Ano	–	Identifikátor existujícího kanálu vytvořeného předchozím voláním funkce <b>/api/log/subscribe</b> .

Parametr	Typ	Povinný	Výchozí hodnota	Popis
<b>timeout</b>	uint32	Ne	0	Specifikuje zpoždění odpovědi (v sekundách) v případě, že fronta kanálu je prázdná. Výchozí hodnota 0 znamená, že interkom odpovídá vždy okamžitě bez jakéhokoliv zpoždění.

Odpověď je ve formátu **application/json** a obsahuje seznam událostí.

Parametr	Typ	Popis
<b>events</b>	array	Pole objektu události. V případě, že během nastaveného timeoutu nenastala žádná událost, je pole prázdné.

**Příklad:**

```
GET /api/log/pull
{
  "success" : true,
  "result" : {
    "events" : [
      {
        "id" : 1,
        "tzShift" : 0,
        "utcTime" : 1437987102,
        "upTime" : 8,
        "event" : "DeviceState",
        "params" : {
          "state" : "startup"
        }
      },
      {
        "id" : 3,
        "tzShift" : 0,
        "utcTime" : 1437987105,
        "upTime" : 11,
        "event" : "RegistrationStateChanged",
        "params" : {
          "sipAccount" : 1,
          "state" : "registered"
        }
      }
    ]
  }
}
```

## Události

Každá událost v poli **events** obsahuje následující informace, které jsou společné pro všechny typy událostí:

Parametr	Typ	Popis
<b>id</b>	uint32	Interní ID záznamu o události (32bit číslo, 1 po restartu interkomu, inkrementované s každou novou událostí)
<b>utcTime</b>	uint32	Absolutní čas vzniku události (Unix Time, UTC – koordinovaný světový čas).
<b>upTime</b>	uint32	Relativní čas vzniku události (počet sekund od restartu interkomu).
<b>tzShift</b>	int32	Rozdíl mezi místním časem a časem UTC v minutách. Přičtením této hodnoty k utcTime se získá místní čas vzniku události dle nastavení časové zóny v zařízení: $localTime = utcTime + tzShift * 60$
<b>event</b>	string	Typ události ("KeyPressed", "InputChanged", ...)
<b>params</b>	object	Specifické parametry události.

## Událost DeviceState

Signalizuje změny stavu zařízení.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	Signalizovaný stav zařízení: <b>"startup"</b> – generováno jednorázově po startu zařízení (vždy úplně první událost)

### Příklad:

```
{
  "id" : 1,
  "tzShift" : 0,
  "utcTime" : 1437987102,
  "upTime" : 8,
  "event" : "DeviceState",
  "params" : {
    "state" : "startup"
  }
}
```

## Událost AudioLoopTest

Signalizuje provedení automatického audio loop testu a jeho výsledek. Událost je signalizovaná vždy po provedení automatického testu (naplánovaného příp. manuálně spuštěného).

Parametr	Typ	Popis
<b>result</b>	string	Výsledek provedeného testu. <b>"passed"</b> – test byl úspěšně proveden a nebyl zjištěn žádný problém <b>"failed"</b> – test byl proveden, ale byl detekován problém s reproduktorem nebo mikrofonom v zařízení

### **Příklad:**

```
{
  "id" : 26,
  "tzShift" : 0,
  "utcTime" : 1438073190,
  "upTime" : 9724,
  "event" : "AudioLoopTest",
  "params" : {
    "result" : "passed"
  }
}
```

## Událost MotionDetected

Signalizuje detekci pohybu pomocí kamery. Událost je dostupná pouze na modelech vybavených kamerou. Událost se generuje pouze v případě, že v konfiguraci kamery interkomu je tato funkce povolena.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	Stav detektoru pohybu: <b>"in"</b> – signalizuje začátek intervalu, ve kterém byl detekován pohyb <b>"out"</b> – signalizuje konec intervalu, ve kterém byl detekován pohyb

### Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 1,
  "event" : "MotionDetected",
  "params" : {
    "state" : "in"
  }
}
```



## Událost NoiseDetected

Signalizuje zvýšenou hladinu zvuku detekovanou pomocí zabudovaného nebo externího mikrofону. Událost se generuje pouze v případě, že v konfiguraci interkomu je tato funkce povolena.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	Stav detektoru hluku: <b>"in"</b> – signalizuje začátek intervalu, ve kterém byl detekován hluk <b>"out"</b> – signalizuje konec intervalu, ve kterém byl detekován hluk

### Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 1,
  "event" : "NoiseDetected",
  "params" : {
    "state" : "in"
  }
}
```

## Události KeyPressed a KeyReleased

Signalizuje stisk (**KeyPressed**) nebo uvolnění (**KeyReleased**) tlačítka nebo klávesy numerické klávesnice.

### Parametry události:

Parametr	Typ	Popis
<b>key</b>	string	Kód stisknutého nebo uvolněného tlačítka: "0" až "9" – tlačítka numerické klávesnice "%1" – "%150" - tlačítka rychlé volby "*" – tlačítko se symbolem * příp. telefonu "#" – tlačítko se symbolem # příp. klíčku

### Příklad:

```
{
  "id" : 4,
  "tzShift" : 0,
  "utcTime" : 1437987888,
  "upTime" : 794,
  "event" : "KeyPressed",
  "params" : {
    "key" : "5"
  }
}
```

## Událost CodeEntered

Signalizuje zadání kódu uživatelem na numerické klávesnici interkomu. Událost se generuje pouze na zařízeních vybavených numerickou klávesnicí.

### Parametry události:

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikáté zadání kódu se jedná.
<b>direction</b>	string	Směr průchodu: <b>"in"</b> – příchod <b>"out"</b> – odchod <b>"any"</b> – průchod <i>Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>code</b>	string	Uživatelem zadaný kód, např. "1234". Kódy mají minimálně dvě číslice a kód <b>"00"</b> nelze použít.
<b>type</b>	string	
<b>uuid</b>	string	Unikátní ID uživatele.
<b>valid</b>	boolean	Platnost zadaného kódu (tj. zda je kód zaveden v konfiguraci interkomu jako platný kód uživatele nebo jako platný univerzální kód spínače). <b>false</b> – kód není platný <b>true</b> – kód je platný

**Příklad:**

```
{
  "id" : 128,
  "tzShift" : 0,
  "utcTime" : 1548078453,
  "upTime" : 1061,
  "event" : "CodeEntered",
  "params" : {
    "ap" : 0,
    "session" : 8,
    "direction" : "in",
    "code" : "1234",
    "type" : "user",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

## Událost CardEntered

Signalizuje přiložení RFID karty ke čtečce. Událost je dostupná pouze na modelech vybavených čtečkou RFID karet.

### Parametry události:

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikáté přiložení karty se jedná.
<b>direction</b>	string	Směr průchodu RFID čtečky: <b>"in"</b> – příchod <b>"out"</b> – odchod <b>"any"</b> – průchod <i>Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>reader</b>	string	Jméno RFID čtečky příp. Wiegand modulu, příp. jedna z následujících hodnot pro nemedulární modely interkomu: <b>"internal"</b> – interní čtečka (2N IP interkomy) <b>"external"</b> – externí čtečka připojená pomocí Wiegand interface <i>Pozn.: Jméno čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>uid</b>	string	Jednoznačný identifikátor přiložené karty (číslo v hexadecimálním formátu, 6–16 znaků, podle typu karty).
<b>uuid</b>	string	Unikátní ID uživatele.
<b>valid</b>	boolean	Platnost přiložené RFID karty (tj. zda je uid karty přiřazeno jednomu z uživatelů v adresáři interkomu). <b>false</b> – karta není platná <b>true</b> – karta je platná

**Příklad:**

```
{
  "id" : 60,
  "tzShift" : 0,
  "utcTime" : 1548078014,
  "upTime" : 622,
  "event" : "CardEntered",
  "params" : {
    "ap" : 0,
    "session" : 5,
    "direction" : "in",
    "reader" : "ext2",
    "uid" : "4BD9E903",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

## Události InputChanged a OutputChanged

Signalizuje změnu stavu logického vstupu (**InputChanged**) nebo výstupu (**OutputChanged**). Aktuální seznam dostupných vstupů a výstupů lze zjistit pomocí funkce `/api/io/caps`.

### Parametry události:

Parametr	Typ	Popis
<b>port</b>	string	Název I/O portu.
<b>state</b>	boolean	Aktuální logický stav I/O portu: <b>false</b> – neaktivní, log.0 <b>true</b> – aktivní, log.1

### Příklad:

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1437987103,
  "upTime" : 9,
  "event" : "OutputChanged",
  "params" : {
    "port" : "led_secured",
    "state" : false
  }
}
```

## Událost SwitchStateChanged

Signalizuje změnu stavu spínače (viz konfigurace interkomu Hardware | Spínače).

### Parametry události:

Parametr	Typ	Popis
<b>switch</b>	uint32	Číslo spínače 1..4
<b>state</b>	boolean	Aktuální logický stav spínače: <b>false</b> – neaktivní, log.0 <b>true</b> – aktivní, log.1
<b>originator</b>		Informuje, jakým způsobem byl switch aktivován. <b>profile</b> – vyvoláno přechodem na nastavený aktivní časový profil. <b>api</b> – vyvoláno pomocí http api (api/switch/ctrl). <b>ap</b> – autentizace uživatele na přístupovém bodu. Událost pak doplňuje číslo app a session. <b>rex</b> – stisk odchodového tlačítka (tlačítko, které otevírá dveře po definovanou dobu, kdy osoba opouští místnost). <b>idt</b> – vyvoláno pomocí http api (api/switch/ctrl), pokud se použila speciální autentizace pro 2N <sup>®</sup> Indoor Touch 2.0, 1.0. <b>dtmf</b> – použití dtmf kódu v hovoru. <b>auth</b> – autorizace pomocí uživatelského, universálního nebo zónového kódu. <b>uni</b> – autorizace pomocí univerzálního kódu. <b>zone</b> – autorizace pomocí zónového kódu. <b>automation</b> – vyvoláno pomocí akce automation.



**Příklad:**

```
{
  "id" : 2,
  "tzShift" : 0,
  "utcTime" : 1437987103,
  "upTime" : 9,
  "event" : "SwitchStateChanged",
  "params" : {
    "switch" : 1,
    "state" : true
  }
}
```

## Událost CallStateChanged

Signalizuje vytvoření, ukončení příp. jinou změnu stavu probíhajícího hovoru.

### Parametry události:

Parametr	Typ	Popis
	string	Specifikuje, zda se jedná o příchozí nebo odchozí hovor: <b>"incoming"</b> – příchozí hovor <b>"outgoing"</b> – odchozí hovor
	string	Aktuální stav automatu hovoru: <b>"connecting"</b> – probíhá sestavování hovoru (pouze odchozí hovory) <b>"ringing"</b> – vyzvánění <b>"connected"</b> – hovor spojen <b>"terminated"</b> – hovor ukončen
<b>peer</b>	string	SIP URI volajícího (příchozí hovory) nebo volaného (odchozí hovory) účastníka.
<b>session</b>	uint32	Jednoznačný identifikátor hovoru. Identifikátor hovoru lze dále použít ve funkcích <b>/api/call/answer</b> , <b>/api/call/hangup</b> a <b>/api/call/status</b> .
<b>call</b>	uint32	TBD

**Příklad:**

```
{
  "id" : 5,
  "tzShift" : 0,
  "utcTime" : 1438064126,
  "upTime" : 660,
  "event" : "CallChanged",
  "params" : {
    "direction" : "incoming",
    "" : "ringing",
    "peer" : "sip:2229@10.0.97.150:5062;user=phone",
    "session" : 1,
    "call" : 1
  }
}
```

## Událost RegistrationChanged

Signalizuje změnu stavu registrace všech SIP účtů k SIP serveru.

### Parametry události:

Parametr	Typ	Popis
<b>sipAccount</b>	uint32	Číslo SIP účtu, na kterém proběhla změna stavu: <b>1</b> – první SIP účet <b>2</b> – druhý SIP účet
	string	Udává nový stav automatu registrace SIP účtu: <b>"registered"</b> – účet byl úspěšně zaregistrován <b>"unregistered"</b> – účet byl odregistrován <b>"registering"</b> – probíhá registrace <b>"unregistering"</b> – probíhá odregistrování účtu

### Příklad:

```
{
  "id" : 3,
  "tzShift" : 0,
  "utcTime" : 1437987105,
  "upTime" : 11,
  "event" : "RegistrationChanged",
  "params" : {
    "sipAccount" : 1,
    "" : "registered"
  }
}
```

## Událost TamperSwitchActivated

Signalizuje aktivaci ochranného spínače – otevření krytu zařízení. Funkce ochranného spínače musí být nakonfigurována v menu Digitální vstupy / Ochranný spínač.

### Parametry události:

Parametr	Typ	Popis
	string	Stav ochranného spínače: <b>"in"</b> – signalizuje aktivaci ochranného spínače (tj. otevření krytu zařízení) <b>"out"</b> – signalizuje deaktivaci ochranného spínače (tj. zavření krytu zařízení)

### Příklad:

```
{
  "id" : 54,
  "tzShift" : 0,
  "utcTime" : 1441357589,
  "upTime" : 158,
  "event" : "TamperSwitchActivated",
  "params" : {
    "state" : "in"
  }
}
```

## Událost UnauthorizedDoorOpen

Signalizuje neautorizované otevření dveří. Vyžaduje připojení snímače otevřených dveří na jeden z digitálních vstupů a příslušné nastavení v menu Digitální vstupy | Stav dveří.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	Stav neautorizovaného otevření dveří: <b>"in"</b> – signalizuje zahájení stavu neautorizovaného otevření <b>"out"</b> – signalizuje ukončení stavu neautorizovaného otevření dveří

### Příklad:

```
{
  "id" : 80,
  "tzShift" : 0,
  "utcTime" : 1441367842,
  "upTime" : 231,
  "event" : "UnauthorizedDoorOpen",
  "params" : {
    "state" : "in"
  }
}
```

## Událost DoorOpenTooLong

Signalizuje dlouhé otevření dveří, resp. nezavření dveří do nastavené doby. Vyžaduje připojení snímače otevřených dveří na jeden z digitálních vstupů a příslušné nastavení v menu Digitální vstupy | Stav dveří.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	Stav dlouhého otevření dveří: <b>"in"</b> – signalizuje zahájení stavu dlouhého otevření dveří. <b>"out"</b> – signalizuje ukončení stavu dlouhého otevření dveří.

### Příklad:

```
{
  "id" : 96,
  "tzShift" : 0,
  "utcTime" : 1441369745,
  "upTime" : 275,
  "event" : "DoorOpenTooLong",
  "params" : {
    "state" : "out"
  }
}
```

## Událost LoginBlocked

Signalizuje dočasné zablokování přístupu k webovému rozhraní z důvodu opakovaného zadání neplatného přihlašovacího jména nebo hesla.

### Parametry události:

Parametr	Typ	Popis
<b>address</b>	string	IP adresa, ze které bylo provedeno opakované přihlášení s neplatnými údaji.

### Příklad:

```
{
  "id" : 5,
  "tzShift" : 0,
  "utcTime" : 1441369745,
  "upTime" : 275,
  "event" : "LoginBlocked",
  "params" : {
    "address" : "10.0.23.32"
  }
}
```



## Událost UserAuthenticated

Signalizuje autentizaci uživatele a následné otevření dveří.

### Parametry události:

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikátou autorizaci uživatele se jedná.
<b>name</b>	string	Specifikuje jméno uživatele z telefonního seznamu
<b>uuid</b>	string	Unikátní ID uživatele.
<b>apbBroken</b>	string	Platnost přiložené karty v Anti-passback oblasti. <b>false</b> – oblast soft APB není aktivní <b>true</b> – oblast soft ABP je aktivní a je porušena

### Příklad:

```
{
  "success" : true,
  "result" : {
    "events" : [
      {
        "id" : 65,
        "tzShift" : 0,
        "utcTime" : 1593606655,
        "upTime" : 7951,
        "event" : "UserAuthenticated",
        "params" : {
          "ap" : 0,
          "session" : 6,
          "name" : "Alice Gruberov\u00E1",
          "uuid" : "8fa29ebc-2fe8-4a8c-9a3b-d8b0351fb6f8",
          "apbBroken" : true
        }
      }
    ]
  }
}
```

## Událost CardHeld

Signalizuje přiložení RFID karty ke čtečce delší než 4 s.

**Parametry události:**

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikáté přiložení karty se jedná.
<b>direction</b>	string	Směr průchodu RFID čtečky: <b>"in"</b> – příchod <b>"out"</b> – odchod <b>"any"</b> – průchod <i>Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>reader</b>	string	Identifikace čtečky, která kartu načetla.
<b>uid</b>	string	Identifikátor uživatele.
<b>valid</b>	string	Platnos přiložené karty (tj. zda je karta zavedena v konfiguraci interkomu jako platná pro uživatele.). <b>false</b> – karta není platná <b>true</b> – karta je platná

**Příklad:**

```
{
  "id" : 9,
  "tzShift" : 0,
  "utcTime" : 1516893493,
  "upTime" : 354,
  "event" : "CardHeld",
  "params" : {
    "ap" : 1,
    "session" : 4,
    "direction" : "out",
    "reader" : "ext2",
    "uid" : "3F00F318E7",
    "valid" : true
  }
}
```

## Událost SilentAlarm

Signalizuje aktivaci tichého alarmu.

### Parametry události:

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikátý SilentAlarm se jedná.
<b>name</b>	string	Specifikuje jméno uživatele z telefonního seznamu.
<b>uuid</b>	string	Unikátní ID uživatele.

### Příklad:

```
{
  "id" : 200,
  "tzShift" : 0,
  "utcTime" : 1548079445,
  "upTime" : 2053,
  "event" : "SilentAlarm",
  "params" : {
    "ap" : 0,
    "session" : 17,
    "name" : "Joseph",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c"
  }
}
```

## Událost AccessLimited

Signalizuje odmítnutí zadaného uživatele.

### Parametry události:

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>type</b>	string	card, code, finger
<b>state</b>	string	Stav, možné hodnoty: in = aktivní, out = neaktivní.

### Příklad:

```
{
  "id" : 408,
  "tzShift" : 0,
  "utcTime" : 1517302112,
  "upTime" : 408951,
  "event" : "AccessLimited",
  "params" : {
    "ap" : 0,
    "type" : "card",
    "state" : "in"
  }
}
```

## Událost PairingStateChange

Signalizuje párování s bluetooth rozhraním.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	pending
<b>authId</b>	string	Autorizační ID

### Příklad:

```
{
  "id" : 197,
  "tzShift" : 0,
  "utcTime" : 1516894499,
  "upTime" : 1360,
  "event" : "PairingStateChanged",
  "params" : {
    "state" : "pending",
    "authId" : "F2CAE955C9B4E81CD00E3A096E52543B"
  }
}
```

## Událost SwitchesBlocked

Signalizuje zablokování zámek ochranným spínačem. Pokud je funkce povolena, po aktivaci ochranného spínače dojde k zablokování všech spínačů po dobu 30 minut. Blokování bude aktivní i po restartu zařízení.

### Parametry události:

Parametr	Typ	Popis
state	string	in, out

### Příklad:

```
{
  "id" : 205,
  "tzShift" : 0, "utcTime" : 1516894667,
  "upTime" : 1528,
  "event" : "SwitchesBlocked",
  "params" : {
    "state" : "in"
  }
}
```

## Událost FingerEntered

Signalizuje přiložení prstu ke čtečce otisků prstů.

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikáté přiložení prstu se jedná.
<b>direction</b>	string	Směr průchodu čtečky otisků prstů: <b>"in"</b> – příchod <b>"out"</b> – odchod <b>"any"</b> – průchod <i>Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>uuid</b>	string	Unikátní ID uživatele.
<b>valid</b>	string	Platnost otisku prstu (tj. zda je otisk prstu zaveden v konfiguraci interkomu jako platný otisk prstu uživatele). <b>false</b> – otisk prstu není platný <b>true</b> – otisk prstu je platný

**Příklad:** Načtení prstu zadaného uživatele

```
{
  "id" : 18,
  "tzShift" : 0,
  "utcTime" : 1548077595,
  "upTime" : 203,
  "event" : "FingerEntered",
  "params" : {
    "ap" : 0,
    "session" : 2,
    "direction" : "in",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```



**Neúspěšné zadání:** Načtení prstu nezadaného uživatele

```
{
  "id" : 1368,
  "tzShift" : 0,
  "utcTime" : 1548145535,
  "upTime" : 62598,
  "event" : "FingerEntered",
  "params" : {
    "ap" : 0,
    "session" : 1,
    "direction" : "in",
    "valid" : false
  }
}
```

## Událost MobKeyEntered

Signalizuje autorizaci pomocí bluetooth čtečky.

**Parametry události:**

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikátou autorizaci pomocí Mobile KEY se jedná.
<b>direction</b>	string	Směr průchodu: <b>"in"</b> – příchod <b>"out"</b> – odchod <b>"any"</b> – průchod <i>Pozn.: Směr průchodu čtečky se nastavuje pomocí konfiguračního rozhraní interkomu.</i>
<b>authid</b>	string	Mobile Key ID.
<b>uuid</b>	string	Unikátní ID uživatele.
<b>valid</b>	string	Platnost Mobile Key (tj. zda je Mobile Key zaveden v konfiguraci interkomu jako platný Mobile Key pro uživatele). <b>false</b> – Mobile Key není platný <b>true</b> – Mobile Key je platný

**Příklad:**

```
{
  "id" : 161,
  "tzShift" : 0,
  "utcTime" : 1548079174,
  "upTime" : 1782,
  "event" : "MobKeyEntered",
  "params" : {
    "ap" : 0,
    "session" : 9,
    "direction" : "in",
    "authid" : "48c48155eed7ea1dbb0b4d534b7459b9",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "valid" : true
  }
}
```

## Událost DoorStateChanged

Signalizuje změnu stavu dveří.

### Parametry události:

Parametr	Typ	Popis
<b>state</b>	string	opened, closed

### Příklad:

```
{
  "id" : 240,
  "tzShift" : 0,
  "utcTime" : 1516895295,
  "upTime" : 2156,
  "event" : "DoorStateChanged",
  "params" : {
    "state" : "opened"
  }
}
```

## Událost UserRejected

Signalizuje odmítnutí autorizace uživatele.

### Parametry události:

Parameter	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikáté odmítnutí autorizace se jedná.
<b>name</b>	string	Jméno uživatele.
<b>uid</b>	string	Unikátní ID uživatele.
<b>reason</b>	string	accessBlocked, switchLocked, invalidTime, invalidProfile, invalidSequence, invalidCredential, authInterrupted, timeout, switchDisabled

**Příklad:**

```
{
  "id" : 173,
  "tzShift" : 0,
  "utcTime" : 1548079274,
  "upTime" : 1882,
  "event" : "UserRejected",
  "params" : {
    "ap" : 0,
    "session" : 10,
    "name" : "Joseph",
    "uuid" : "54877b0e-4cc3-c645-9530-6c7850f47a9c",
    "reason" : "invalidCredential"
  }
}
```

## Událost DisplayTouched

Signalizuje dotyk na displeji.

### Parametry události:

Parametr	Typ	Popis
<b>x</b>	string	Souřadnice bodu dotyku na displeji. Maximální hodnota je dána rozlišením displeje.
<b>y</b>	string	Souřadnice bodu dotyku na displeji.
<b>dx</b>	string	Změna souřadnic při posunu na displeji, může nabývat i záporné hodnoty. Maximální hodnota je dána rozlišením displeje.
<b>dy</b>	string	Změna souřadnic při posunu na displeji.

### Příklad:

```
{
  "id" : 337,
  "tzShift" : 0,
  "utcTime" : 1517301424,
  "upTime" : 408263,
  "event" : "DisplayTouched",
  "params" : {
    "x" : 89,
    "y" : 100,
    "dx" : 0,
    "dy" : 0
  }
}
```

## Událost DtmfEntered

Signalizuje DTMF kód v hovoru.

```
{
  "id" : 86,
  "tzShift" : 0,
  "utcTime" : 1558522871,
  "upTime" : 3531,
  "event" : "DtmfEntered",
  "params" : {
    "code" : "00",
    "type" : "uni",
    "call" : 3,
    "valid" : true
  }
}
```

Parametr	Typ	Popis
<b>code</b>	string	Znění zadaného kódu.
<b>type</b>	string	Informuje, jaký typ kódu byl použit. <b>uni</b> – univerzální kód spínače <b>user</b> – uživatelský kód
<b>call</b>	string	ID hovoru.
<b>valid</b>	string	Platnost zadaného kódu (tj. platný univerzální kód spínače nebo platný uživatelský kód). <b>false</b> – kód není platný <b>true</b> – kód je platný



## Událost AccessTaken

Signalizuje přiložení karty v Anti-passback oblasti.

```
{
  "success" : true,
  "result" : {
    "events" : [

    ]
  }
}
```

## Událost ApLockStateChanged

Signalizuje změnu stavu (zapnuto/vypnuto) nouzového uzamčení.

```
{
  "id" : 35,
  "tzShift" : 0,
  "utcTime" : 1558522465,
  "upTime" : 3125,
  "event" : "ApLockStateChanged",
  "params" : {
    "ap" : 0,
    "state" : "in"
  }
}
```

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>state</b>	string	Stav změny stavu. <b>"in"</b> – signalizuje začátek intervalu nouzového uzamčení <b>"out"</b> – signalizuje konec intervalu nouzového uzamčení

## Událost RexActivated

Signalizuje aktivaci vstupu, která je nastavena na tlačítko REX.

```
{
  "id" : 29,
  "tzShift" : 0,
  "utcTime" : 1558522162,
  "upTime" : 2822,
  "event" : "RexActivated",
  "params" : {
    "ap" : 1,
    "session" : 1
  }
}
```

Parametr	Typ	Popis
<b>ap</b>	string	Access Point, možné stavy: 0 = entry, 1 = exit.
<b>session</b>	string	Informuje, o kolikátou aktivaci REX tlačítka se jedná.

## Událost LiftStatusChanged

Signalizuje detekci připojení/odpojení Lift Control modulu.

```
{
  "id" : 2871,
  "tzShift" : 0,
  "utcTime" : 1561540370,
  "upTime" : 73822,
  "event" : "LiftStatusChanged",
  "params" : {
    "module" : 0,
    "ready" : true
  }
},
```

Parametr	Typ	Popis
<b>module</b>	string	Informuje o ID modulu.
<b>ready</b>	string	Informuje o stavu modulu. <b>false</b> – odpojeno <b>true</b> – připojeno

## Událost LiftFloorsEnabled

Signalizuje permanentní povolení přístupu na patro nebo trvalý přístup uživatele

```
{
  "id" : 2850,
  "tzShift" : 0,
  "utcTime" : 1561540011,
  "upTime" : 73463,
  "event" : "LiftFloorsEnabled",
  "params" : {
    "type" : "user"
    "floors" : [
      0, 1, 2, 3, 4
    ],
    "uuid" : "621a5a49-1f8b-d34c-9a8b-881055864deb",
  }
},
```

```
{
  "id" : 2855,
  "tzShift" : 0,
  "utcTime" : 1561540016,
  "upTime" : 73468,
  "event" : "LiftFloorsEnabled",
  "params" : {
    "type" : "public"
    "floors" : [
      1, 4
    ],
  }
},
```

Parametr	Typ	Popis
<b>type</b>	string	Informuje, o jaký typ přístupu se jedná. <b>public</b> – změna veřejného přístupu <b>user</b> – při autentizaci uživatele
<b>floors</b>	string	Informuje o přístupných patrech.

## Událost LiftConfigChanged

Signalizuje změnu nastavení řízení výtahu.

```
{
  "id" : 2860,
  "tzShift" : 0,
  "utcTime" : 1561540163,
  "upTime" : 73615,
  "event" : "LiftConfigChanged",
  "params" : {
    "hash" : 11
  }
},
```

Parametr	Typ	Popis
<b>hash</b>	string	Unikátní číslo změny konfigurace.

## Událost CapabilitiesChanged

Signalizuje změnu dostupných funkcí.

```
{
  "success": true,
  "result": {
    "events": [
      {
        "id": 21,
        "tzShift": 0,
        "utcTime": 1585037151,
        "upTime": 256,
        "event": "CapabilitiesChanged",
        "params": {

        }
      }
    ]
  }
}
```

Parametr	Typ	Popis
<b>id</b>	string	Pořadové číslo události.
<b>tzShift</b>	uint32	Rozdíl mezi místním časem a časem UTC v minutách. Přičtením této hodnoty k utcTime se získá místní čas vzniku události dle nastavení časové zóny v zařízení: $localTime = utcTime + tzShift * 60$
<b>utcTime</b>	uint32	Absolutní čas vzniku události (Unix Time, UTC – koordinovaný světový čas).
<b>upTime</b>	uint32	Relativní čas vzniku události (počet sekund od restartu interkomu).
<b>event</b>	string	Typ události CapabilitiesChanged.
<b>params</b>	object	Specifické parametry události.

## 5.11 api audio

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/audio**.

- [5.11.1 api audio test](#)

### 5.11.1 api audio test

Funkce **/api/audio/test** spustí automatický test zabudovaného mikrofону a reproduktoru interkomu. Po provedení testu je výsledek zapsán do logu v zařízení jako událost **AudioLoopTest**.

Funkce je součástí služby **Audio** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Audio (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

#### **Příklad:**

```
GET /api/audio/test
{
  "success" : true
}
```

## 5.12 api email

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/email**.

- [5.12.1 api email send](#)

### 5.12.1 api email send

Funkce **/api/email/send** odešle ze zařízení e-mail na uvedenou adresu. Pro správnou funkci je potřeba nakonfigurovat službu SMTP v zařízení (tj. nastavit adresu SMTP serveru, správné přihlašovací údaje apod.)

Funkce je součástí služby **E-mail** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **E-mail (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Parametry požadavku:

Parametr	Popis
<b>to</b>	Povinný parametr obsahující e-mailovou adresu pro doručení.
<b>subject</b>	Povinný parametr specifikující předmět zprávy.
<b>body</b>	Nepovinný parametr specifikující obsah zprávy (může obsahovat html značky). Pokud parametr není uveden, zpráva bude doručena bez obsahu.
<b>pictureCount</b>	Nepovinný parametr specifikující počet přiložených obrázků z kamery. V případě, že není uveden, obrázky nejsou přiloženy. Parametr může nabývat hodnot <0, 5>.
<b>timeSpan</b>	Časový rozsah snímků přiložených k emailu v sekundách. Parametr je nepovinný, implicitní hodnota 0.
<b>width</b>	Nepovinné parametry nastavující rozlišení přiložených obrázků z kamery. Výška a šířka snímku musí odpovídat jedné z podporovaných variant (viz funkce <b>api/camera/caps</b> ).
<b>height</b>	

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.



**Příklad:**

```
GET /api/email/send?to=somebody@email.com&subject=Hello&body=Hello
{
  "success" : true
}
```

## 5.13 api pcap

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/pcap**.

- [5.13.1 api pcap](#)
- [5.13.2 api pcap restart](#)
- [5.13.3 api pcap stop](#)
- [5.13.4 api pcap live](#)
- [5.13.5 api pcap live stop](#)
- [5.13.6 api pcap live stats](#)

### 5.13.1 api pcap

Funkce **/api/pcap** slouží ke stažení zaznamenaného provozu na síťovém rozhraní zařízení (pcap soubor). Záznam síťového provozu lze také řídit pomocí funkcí **/api/pcap/restart** a **/api/pcap/stop**.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď ve formátu **application/pcap** a stažený soubor lze přímo otevřít např. v programu Wireshark.

**Příklad:**

```
GET /api/pcap
```

### 5.13.2 api pcap restart

Funkce **/api/pcap/restart** provede odstranění všech zaznamenaných paketů a znovu spustí záznam provozu na síťovém rozhraní zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/pcap/restart
{
  "success" : true
}
```

### 5.13.3 api pcap stop

Funkce **/api/pcap/stop** zastaví zaznamenávání provozu na síťovém rozhraní zařízení.

Funkce je součástí služby **System** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **System (řízení)**.

Pro tuto funkci lze použít metody **GET** nebo **POST**.

Funkce nemá žádné parametry.

Odpověď je ve formátu **application/json** a neobsahuje žádné parametry.

**Příklad:**

```
GET /api/pcap/restart
{
  "success" : true
}
```

### 5.13.4 api pcap live

Funkce **api/pcap/live** slouží ke spuštění zachytávání chunk paketů.

#### Skupiny služeb privilegií

- Služba je System.
- Privilegia jsou System – řízení.

#### Metody

- GET
- POST

## Požadavek

Požadavek obsahuje parametry ve formátu URL (nebo **application/x-www-form-urlencoded** při použití metody POST).

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>duration</b>	Ne	Celé číslo, které definuje délku stahování v sekundách	doba neurčitá	Definuje dobu trvání zachytávání paketů. Je-li parametr vynechán nebo roven 0, doba trvání je neurčitá (tj. dokud není stahování zastaveno funkcí <b>api/pcap/live/stop</b> nebo ukončeno cílovým zařízením).

## Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live?duration=10
```

## Odpověď

Zařízení začne streamovat chunky po úspěšném požadavku.

## Příklad použití jazyka Python pro stažení zachycených paketů

```
command = requests.post( "https://" + address + "/api/pcap/live?duration=120",
verify=False, stream=True, auth=HTTPBasicAuth("admin", "pass") ) with
open("trace.pcap", 'wb') as f: for chunk in command.iter_content(chunk_size=None):
f.write(chunk)
```

Pokud jedno zachytávání paketů už probíhá, není možné spustit další.

## 5.13.5 api pcap live stop

Funkce **api/pcap/live/stop** slouží k zastavení zachytávání chunk paketů.

## Skupiny služeb privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

## Metody

- GET
- POST

## Požadavek

Požadavek nemá žádné parametry.

## Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live/stop
```

## Odpověď

Zařízení přestane streamovat chunky po úspěšném požadavku. Požadavek slouží k zastavení zachytávání bez zadané doby trvání nebo předčasnému zastavení zachytávání s danou dobou trvání.

Zařízení zašle v odpovědi **success : true**, i když neprobíhá žádné zachytávání. Pro tento koncový bod nejsou žádné konkrétní chybové kódy.

## 5.13.6 api pcap live stats

Funkce **api/pcap/live/stats** slouží k získání statusu zachytávání chunk paketů.

## Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

## Metody

- GET
- POST

## Požadavek

Požadavek nemá žádné parametry.

## Příklad požadavku

```
URL: https://192.168.1.1/api/pcap/live/stats
```

### Odpověď

Odpověď je ve formátu **application/json**. Obsahuje klíče **success** a **result**. Hodnota klíče **result** obsahuje detailní informace o stavu zachytávání paketů.

Tabulka 1. Klíče JSON odpovědi

Klíč	Typické vrácené hodnoty	Popis
<b>running</b>	true nebo false	Oznamuje, zda zachytávání chunk paketů právě probíhá, nebo neprobíhá.
<b>bytesSent</b>	Celé číslo	Obsahuje počet bytů poslaných v právě probíhající zachytávání otevřených paketů. Jestliže zachytávání paketů právě neprobíhá, hodnota je 0.
<b>packetsSent</b>	Celé číslo	Obsahuje počet paketů poslaných v právě probíhající zachytávání otevřených paketů. Jestliže zachytávání paketů právě neprobíhá, hodnota je 0.

### Příklad odpovědi

```
{ "success": true, "result": { "running": true, "bytesSent": 11261, "packetsSent": 90 } }
```

## 5.14 api dir

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/dir**.

- [5.14.1 api dir template](#)
- [5.14.2 api dir create](#)
- [5.14.3 api dir update](#)
- [5.14.4 api dir delete](#)
- [5.14.5 api dir get](#)
- [5.14.6 api dir query](#)

### 5.14.1 api dir template

Funkce **/api/dir/template** umožňuje vyhledávat šablonu záznamu v adresáři.

## Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

## Metody

- GET
- POST

## Požadavek

Tabulka 1. Parametry požadavku

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
není	-	-	-	-

## Příklad požadavku

```
https://192.168.1.1/api/dir/template
```

## Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole s jedním objektem (šablona záznamu), který obsahuje všechny dostupné klíče záznamu v adresáři včetně výchozích hodnot pro dané zařízení.

### ✓ Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

**Note**

- Dostupnost klíčů závisí na modelu, typu a hardwarové konfiguraci zařízení (např. klíč photo je dostupný pouze u zařízení, která jsou vybavena displejem a ukládají obrázky do adresáře).

Tabulka 2. Klíče odpovědi JSON v poli **users**

Klíč	Typické vrácené hodnoty	Popis
<b>uuid</b>	Prázdný	Unikátní uživatelský identifikátor. Když je pomocí funkce <b>api/dir/create</b> vytvořen nový záznam v adresáři, je jeho uuid buď ve formě parametru požadavku, nebo je automaticky generován zařízením.  Formát uuid je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid.
<b>deleted</b>	false	Indikuje, jestli byl záznam v adresáři vymazán, nebo ne. Vymazané záznamy zůstávají v adresáři, dokud není dosažen maximální počet záznamů. Uložené vymazané záznamy si ponechávají uuid jen z důvodů ukládání do logů. Jsou dvě platné hodnoty: false, true.
<b>owner</b>	Prázdný	Informace o tom, zda je záznam v adresáři vzdáleně spravován aplikací 2N® My2N nebo 2N® Access Commander nebo jiným systémem pro vzdálenou správu. Tato hodnota je určena pro interní potřeby 2N® TELEKOMUNIKACE a.s., případně pro vymazání skupiny uživatelů (viz <b>api/dir/delete</b> ).
<b>name</b>	Prázdný	Název záznamu v adresáři (název uživatele nebo zařízení). Očekává se řetězec o maximální délce 63 znaků. Název může zůstat nevyplněn (v takovém případě je záznam v logu, e-mailu apod. identifikován pomocí uuid).



Klíč	Typické vrácené hodnoty	Popis
<b>photo</b>	Prázdný	Obrázek záznamu v adresáři (např. fotografie uživatele nebo logo společnosti). Ukládá se jako jpeg soubor šifrovaný pomocí base64 s touto syntaxí: <b>PŘÍKLAD: <i>_DATA_IN_BASE64</i></b>
<b>email</b>	Prázdný	E-mailová adresa záznamu v adresáři. Očekávaný formát je namestructure@domainhierarchy.top. Je možno zadat i více adres a oddělit je čárkou (uložit je jako řetězec).

Klíč	Typické vrácené hodnoty	Popis
<b>treepath</b>	/	<p>Definice pozic záznamu v adresáři na displeji.</p> <ul style="list-style-type: none"> <li>• Výchozí pozice je kořenová složka. Tuto pozici zajistíte pouhým stiskem jednoho lomítka. <b>PŘÍKLAD:</b> / ukázka záznamu v kořenové složce</li> <li>• Záznam je možno umístit na několik pozic a ty oddělit středníkem (;). <b>PŘÍKLAD:</b> <b>/Folder1;/Folder2/</b> ukazuje záznam ve složce Folder1 i Folder2</li> <li>• Záznamu může být přidělena volací skupina, která slouží i jako náhradní název v určité pozici na displeji, když se vynechá lomítko na konci definice pozice. <b>PŘÍKLAD:</b> <b>/Folder1/Calling Group</b> ukazuje záznam ve složce Folder1 pod názvem "Calling Group"</li> <li>• Záznamu je v každé pozici možno individuálně přiřadit prioritu (prioritní záznam se pak zobrazuje nad ostatními záznamy) přidáním <b>:1</b> na konec definice pozice. <b>PŘÍKLAD:</b> <b>/Folder1/:1;Folder2/Calling Group:1</b> ukazuje prioritní záznam ve Folder1 pod jeho názvem a prioritní záznam ve Folder2 pod názvem "Calling Group"</li> <li>• Jedné volací skupině je možno přiřadit více záznamů (výběr této volací skupiny na displeji vede ke skupinovému volání na všechny záznamy ve volací skupině). <b>PŘÍKLAD:</b> <i>Entry1: /Calling Group</i> <i>Entry2: /Calling Group</i> ukazuje oba záznamy v kořenové složce na jednom řádku s názvem "Calling Group" (při výběru tohoto řádku se volá na oba záznamy)</li> </ul>
<b>virtNumber</b>	Prázdný	<p>Virtuální číslo záznamu v adresáři. Virtuální čísla se dají vytáčet na numerické klávesnici (pokud je k dispozici). Maximální délka je 7 znaků. První a poslední znak se dá vybrat z rozmezí A až Z nebo 0 až 9. Zbylé znaky mohou být mezi 0 a 9.</p>

Klíč	Typické vrácené hodnoty	Popis
<b>deputy</b>	Prázdný	Uuid zástupného záznamu, který je volán, když původně volaný uživatel není k dispozici nebo neodpovídá. Jestliže zástupce nemá nastaven zástupný uuid, zůstane klíč prázdný.
<b>buttons</b>	Prázdný	Tlačítka přiřazená tomuto záznamu v adresáři. Pole celých čísel (přiřazovaných podle pozice tlačítka počínaje 1) oddělených čárkami.

Klíč	Typické vrácené hodnoty	Popis
<b>callPos</b>	Pole	<p>Informace o volání pro záznam v adresáři. Zadává se jako pole až tří objektů, tj. tří sad informací o volání. Každý z těchto tří objektů může obsahovat následující klíče:</p> <ul style="list-style-type: none"> <li>• peer - telefonní číslo záznamu v adresáři</li> <li>• profiles - časové profily, pokud je toto telefonní číslo platné (číslo se nevytočí, když je neplatné) <ul style="list-style-type: none"> <li>• P=X,..,Y kde X,..,Y představují pole předdefinovaných časových profilů oddělených čárkou od 0 (časový profil 1) do 19 (časový profil 20) <b>PŘÍKLAD: P=1,3,5</b> znamená, že předdefinované profily 2, 4 a 6 definují dobu platnosti telefonního čísla</li> <li>• D=A,..,B@H:MM-H:MM kde A,..,B představují pole dnů v týdnu oddělených čárkou (0 až 6 pro neděle až sobota, 7 je svátek), H znamená denní hodinu (od 0 do 23), MM znamená minuty (od 00 do 59) - dvě hodnoty definující interval. Jednotlivé definice se dají kombinovat a oddělovat středníkem. <b>PŘÍKLAD: D=7@0:15-13:15;D=5,7@13:15-15:15;D=7@15:15-23:30</b> znamená, že telefonní číslo je platné ve svátek od 0:15 do 13:15, v pátek a ve svátek od 13:15 to 15:15 a ve svátek od 15:15 do 23:30. <b>PŘÍKLAD: D=5,7</b> znamená, že telefonní číslo je platné v pátek a ve svátek celý den.</li> </ul> </li> <li>• grouped - definuje, zda se číslo vytočí ve skupinovém volání spolu s předchozím číslem (třetí číslo se vytočí se zástupcem). Může nabývat hodnot true nebo false.</li> <li>• ipEye - definuje IP adresu počítače, na němž běží aplikace 2N® IP Eye (pro příjem videa, jestliže telefon nemá displej).</li> </ul>

Klíč	Typické vrácené hodnoty	Popis
<b>access</b>	JSON objekt	<p>Informace o kontrole přístupu pro záznam v adresáři. Obsahuje tyto klíče:</p> <ul style="list-style-type: none"> <li>• <b>validFrom</b> - definice začátku doby platnosti pověření záznamu v adresáři. Zadává se ve formátu Unix Time. Je-li pole prázdné, začíná doba platnosti na začátku časového období (tj. 00:00:00 UTC 1.1. 1970).</li> <li>• <b>validTo</b> - definice konce doby platnosti pověření záznamu v adresáři. Zadává se ve formátu Unix Time. Je-li pole prázdné, končí doba platnosti na konci časového období (tj. 03:14:07 UTC 19.1. 2038).</li> <li>• <b>accessPoints</b> - obsahuje pole přístupových bodů (dva přístupové body, vstup a výstup). Každá položka pole je reprezentována JSON objektem s těmito klíči: <ul style="list-style-type: none"> <li>• <b>enabled</b> - definuje, zda je obecně možné tento přístupový bod použít (tj. zda je možné se na tomto konkrétním přístupovém bodu autentizovat). Jsou dvě platné hodnoty: <code>true</code>, <code>false</code>.</li> <li>• <b>profiles</b> - definuje, zda je aktuálně možné tento přístupový bod použít (tj. zda je možné se v tuto chvíli na tomto přístupovém bodě autentizovat). Syntax definice profilu je stejná jako u <code>callPos</code>.</li> </ul> </li> <li>• <b>pairingExpired</b> - určuje, zda párování záznamu v adresáři pomocí Bluetooth vypršelo, nebo ne. Jsou dvě platné hodnoty: <code>true</code>, <code>false</code>.</li> <li>• <b>virtCard</b> - definuje virtuální kartu záznamu v adresáři, která je přenášena na rozhraní Wiegand a do dalších systémů třetích stran, pokud je autentizace záznamu v adresáři úspěšná. Očekává se 6 až 32 hexadecimálních znaků.</li> <li>• <b>card</b> - definuje až dvě karty záznamu v adresáři, které se používají pro autentizaci. Čísla karet jsou zapsána jako pole řetězců. Očekává se 6 až 32 hexadecimálních znaků.</li> <li>• <b>mobkey</b> - definuje ID Bluetoothové autentizace záznamu v adresáři. Očekává se 32 hexadecimálních znaků.</li> </ul>

Klíč	Typické vrácené hodnoty	Popis
		<ul style="list-style-type: none"> <li>• fpt - Šablony otisků prstů záznamu v adresáři. Očekává se zašifrovaný otisk prstu (pro více informací kontaktujte Technickou podporu).</li> <li>• pin - definuje PIN záznamu v adresáři. Očekává se 2 až 15 číslic.</li> <li>• apbException - definuje, zda má záznam v adresáři výjimku ze systému anti-passback (např. pokud ji má, mohou se jeho pověření použít vícenásobně pro vstup bez výstupu). Jsou dvě platné hodnoty: true, false.</li> <li>• code - definuje až tři samostatné kódy spínačů. Ty jsou zapsány v poli řetězců (2 až 15 číslic). První pozice v poli definuje kód prvního spínače. Při vložení prázdného řetězce dojde k přeskočení kódu pro daný spínač.</li> <li>• liftFloors - definuje konfiguraci přístupných pater výtahu po autentizaci. Očekává se následující formát:  F=P,...,R@PROFILE1_DEFINITION   F=X,...,Z@PROFILE2_DEFINITION  kde P,...,R a X,...,Z jsou pole podlaží oddělených čárkou (0 až 63). io_1_1 se zadává jako 0, io_1_5 jako 4, io_2_2 jako 9 a tak dále. Pole pater aktivních v určitých profilech se oddělují  . Dají se použít předem definované profily nebo ad hoc definice nového profilu (viz definice syntaxe v callPos/profilu). Definici profilu lze vynechat, když není použit žádný profil.  <b>PŘÍKLAD: F=2,4</b> definuje patra bez časového profilu (uživatelé jsou vždy přístupná).  <b>PŘÍKLAD: F=5@P=0,7</b> definuje, že šesté patro (F=5) je přístupné v pondělí a ve svátek celý den.  <b>PŘÍKLAD: F=0@P=7,13 </b>  <b>F=0@D=5@9:15-11:45;D=4@11:45-13:30</b> definuje, že první patro (F=0) je přístupné podle předdefinovaných profilů 8 a 14 (P=7,13) a podle časového profilu definovaného v řetězci definice. </li> </ul>

Klíč	Typické vrácené hodnoty	Popis
		<ul style="list-style-type: none"><li>licensePlates - definuje sadu registračních značek přiřazených uživateli (pro otevření dveří na základě rozpoznání registrační značky). Jednotlivé registrační značky se oddělují čárkou. Bílé znaky se ignorují. Maximální povolený počet znaků je 255 (včetně bílých znaků). Maximální počet registračních značek v poli je 20 a každá registrační značka může mít maximálně 10 znaků jiných než bílých.</li></ul>
<b>timestamp</b>	0	Časová značka změn provedených v adresáři. Časové značky se v adresáři generují automaticky ve vzestupném pořadí. Více informací o použití časových značek najdete v kapitole <b>api/dir/query</b> .

## Příklad odpovědi

```
{
  "success": true,
  "result": {
    "series": "5247939846841727056",
    "users": [
      {
        "uuid": "",
        "deleted": false,
        "owner": "",
        "name": "",
        "photo": "",
        "email": "",
        "treepath": "\\",
        "virtNumber": "",
        "deputy": "",
        "buttons": "",
        "callPos": [
          {
            "peer": "",
            "profiles": "",
            "grouped": false,
            "ipEye": ""
          },
          {
            "peer": "",
            "profiles": "",
            "grouped": false,
            "ipEye": ""
          },
          {
            "peer": "",
            "profiles": "",
            "grouped": false,
            "ipEye": ""
          }
        ],
        "access": {
          "validFrom": "0",
          "validTo": "0",
          "accessPoints": [
            {
              "enabled": true,
              "profiles": ""
            },
            {
              "enabled": true,
              "profiles": ""
            }
          ]
        }
      }
    ]
  }
}
```



```

    }
  ],
  "pairingExpired": false,
  "virtCard": "",
  "card": [
    "",
    ""
  ],
  "mobkey": "",
  "fpt": "",
  "pin": "",
  "apbException": false,
  "code": [
    "",
    "",
    "",
    ""
  ],
  "licensePlates": "",
  "liftFloors": ""
},
"timestamp": 0
}
]
}

```

### 5.14.2 api dir create

Funkce **/api/dir/create** umožňuje vytvářet (nebo přepisovat) pole záznamů v adresáři a nastavovat jejich vybraná pole.

#### Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

#### Metody

- PUT

#### Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich objektovém zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>force</b>	Ne	true, false	false	<p>Tento klíč vybírá, zda bude záznam v adresáři identifikovaný daným uuid (viz níže) přepsán novou sadou dat.</p> <p>Když bude hodnota tohoto klíče nastavena na true, přepíše se existující data novou sadou dat a zbylá pole budou nastavena na výchozí hodnoty.</p> <p>Když bude nastavena na false nebo vynechána (omitted) a v adresáři existuje záznam s daným uuid, zařízení odpoví chybovým kódem EDIR_UUID_ALREADY_EXISTS a změny konfigurace nebudou provedeny.</p> <p>Tento klíč nijak neovlivňuje vytvoření nového záznamu v adresáři.</p>
<b>users</b>	Ne	pole JSON objektů definujících parametry záznamu v adresáři	-	<p>Jestliže je klíč vynechán nebo jeho hodnota je prázdné pole, odpověď zařízení obsahuje pouze klíč <b>series</b> (nevytvoří se nové záznamy v adresáři).</p> <p>Do pole je možné zadávat prázdné objekty. Zařízení vytvoří pro každý prázdný objekt prázdný záznam v adresáři (pouze s přiřazeným identifikátorem uuid).</p> <p>Jestliže nějaký objekt v poli obsahuje klíč uuid, vytvoří se nebo změní záznam s daným uuid nebo zařízení odpoví chybovým kódem.</p> <p>Jestliže nějaký objekt v poli neobsahuje uuid, zařízení vytvoří nový záznam a přiřadí mu nový identifikátor uuid. Parametry záznamu jsou nastaveny na hodnoty podle klíčů definovaných v JSON struktuře požadavku. Viz příklad dole.</p> <p>Přehled všech dostupných klíčů ve JSON definici záznamu v adresáři najdete v kapitole <b>api/dir/template</b>.</p>

## Příklad požadavku

```
URL: https://192.168.1.1/api/dir/create JSON { "force": true, "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@def.cz", "access": { "pin": "1234" } }, { "name": "ABCD2", "owner": "My2N", "email": "abcd2@def.cz" }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD3", "email": "something", "access": { "pin": "5678" }, "test": "something", "albert": "einstein" }, {}, {} ] }
```

Pokud v adresáři není žádný záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení vytvoří záznam s tímto identifikátorem a nastaví název jeho parametrů, e-mail a přístup na zadané hodnoty.

Je-li v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení přepíše název jeho parametrů, e-mail a přístup na zadané hodnoty a všechny jeho ostatní parametry nastaví na výchozí hodnoty (protože parametr klíče **force** je nastaven na true).

Zařízení vytvoří druhý záznam, přiřadí mu náhodný identifikátor uuid, nastaví jeho název, vlastníka a e-mail na zadané hodnoty a zbylým parametrům ponechá výchozí hodnoty.

Třetí záznam nepřepíše stávající záznam se stejným uuid, protože je tam několik chyb (e-mail má nesprávný formát, jsou tam neexistující pole s názvem **test** a **albert**).

Navíc se vytvoří dva prázdné nové záznamy (protože jsou v poli dva prázdné objekty). Každému z nich je přidělen náhodný identifikátor uuid, zbylé parametry jsou nastaveny na výchozí hodnoty.

## Odpověď

Odpověď je ve formátu **application/json**. **Výsledek (result)** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

### ✓ Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Klíč	Typické vrácené hodnoty	Popis
<b>uid</b>	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) vytvořeného, upraveného nebo nezměněného záznamu.
<b>timestamp</b>	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole <b>api/dir/query</b> . Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.

Klíč	Typické vrácené hodnoty	Popis
<b>errors</b>	pole chybových objektů	<p>Chybový objekt obsahující pole všech chyb, které nastaly. Objekt <b>errors</b> je uveden, jen když se změna v adresáři nepodařilo provést.</p> <p>V hodnotě klíče <b>code</b> je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři.</p> <p>Další zpřesnění důvodu chyby může být v hodnotě klíče <b>field</b> pro chybové kódy EDIR_FIELD_NAME_UNKNOWN a EDIR_FIELD_VALUE_ERROR, identifikující klíč nebo hodnotu nespĺňující validační pravidla.</p> <p>V odpovědi se mohou vrátit následující chybové kódy:</p> <ul style="list-style-type: none"> <li>• EDIR_UUID_INVALID_FORMAT - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid.</li> <li>• EDIR_UUID_ALREADY_EXISTS - v adresáři existuje záznam se zadaným identifikátorem uuid a klíč <b>force</b> je nastaven na false nebo omitted. Požadovaná změna proto nemůže být provedena.</li> <li>• EDIR_FIELD_NAME_UNKNOWN - neznámý klíč. Seznam dostupných klíčů pro dané zařízení najdete v kapitole <b>/api/dir/template</b>.</li> <li>• EDIR_FIELD_VALUE_ERROR - zadaná hodnota nespĺňuje validační kritéria (hodnota není platná).</li> <li>• EDIRLIM_USER - adresář je plný.</li> <li>• EDIRLIM_PHOTO - byl dosažen maximální počet fotografií v úložišti. Nové záznamy mohou být vytvořeny jen bez fotografií.</li> <li>• EDIRLIM_FPT - byl dosažen maximální počet šablon otisků prstů v úložišti. Nové záznamy mohou být vytvořeny jen bez otisků prstů.</li> <li>• EINCONSISTENT - v hodnotách klíčů <b>validFrom</b> a <b>validTo</b> je nesrovnalost (<b>validFrom</b> musí být menší než <b>validTo</b>).</li> </ul>

## Příklad odpovědi

```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "timestamp": 34 }, { "uuid":
"044197A7-54AD-7577-6EEA-787A6097263E", "timestamp": 35 }, { "errors": [ { "code":
"EDIR_FIELD_VALUE_ERROR", "field": "email" }, { "code": "EDIR_FIELD_NAME_UNKNOWN",
"field": "test" }, { "code": "EDIR_FIELD_NAME_UNKNOWN", "field": "albert" } ] }, {
"uuid": "41970B83-21C8-45DD-8FFC-787A6097263E", "timestamp": 36 }, { "uuid":
"0447BBA7-6E7c-420C-A654-466D43D6A067", "timestamp": 37 } ] } }
```

První záznam je vytvořen se zadaným identifikátorem uuid a zadanými poli (nezadaná pole jsou nastavena na výchozí hodnoty). Záznam se vytvoří bez ohledu na existenci záznamu se stejným uuid, protože klíč **force** je v požadavku nastaven na true. Vrábí se časová značka změny.

Druhý záznam je vytvořen s náhodným uuid a jsou vyplněna zadaná pole (nezadaná pole jsou nastavena na výchozí hodnoty). Vrábí se časová značka změny.

Třetí objekt v požadavku obsahoval neplatný formát e-mailové adresy. Navíc klíče **test** a **albert** odkazovaly k neexistujícím polím.

Vytvoření čtvrtého a pátého záznamu proběhlo úspěšně s náhodně přiřazeným identifikátorem uuid a všechna pole byla nastavena na výchozí hodnoty. Časová značka v zařízení se tudíž dvakrát aktualizovala. Vrábí se časové značky změn.

 **Tip**

- Jestliže klíč **force** v požadavku není nastaven na true, všechny pokusy o vytvoření záznamu se stávajícím uuid skončí chybovým kódem EDIR\_UUID\_ALREADY\_EXISTS.

## 5.14.3 api dir update

Funkce **/api/dir/update** umožňuje aktualizaci pole záznamů v adresáři a nastavení jejich vybraných polí.

**Skupiny služeb a privilegií**

- Služba je System.
- Privilegia jsou Systém – řízení.

**Metody**

- PUT

## Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>users</b>	Ano	pole JSON objektů definujících parametry záznamu v adresáři	-	Pole musí obsahovat nejméně jeden objekt s klíčem <b>uuid</b> , který definuje, který záznam se má aktualizovat.  Přehled dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole <b>api/dir/template</b> .

## Příklad požadavku

```
URL: https://192.168.1.1/api/dir/update JSON { "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@def.cz", "access": { "pin": "1234" } }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654", "name": "ABCD2", "owner": "My2N", "email": "abcd2@def.cz" }, { "uuid": "01234567-89A-CDEF-0123-456789ABCDEF", "name": "ABCD3", "owner": "My2N", "email": "abcd3@def.cz" }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD4", "owner": "My2N", "email": "abcd4@def.cz", "albert": "einstein" }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD4", "owner": "My2N", "email": "abcd4@def.cz", "access.pin": "hello" } ] }
```

Pokud není v adresáři žádný záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, budou jeho parametry aktualizovány podle hodnot zadaných pro jednotlivé klíče. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Třetí záznam nebude aktualizován (uuid má špatný formát).

Čtvrtý záznam nebude aktualizován (neznámý název pole).

Pátý záznam nebude aktualizován (nesprávný formát přístupového PINu).

### Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

#### ✓ Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Klíč	Typické vrácené hodnoty	Popis
<b>uuid</b>	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) upraveného nebo nezměněného zápisu.
<b>timestamp</b>	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole <b>api/dir/query</b> . Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.



Klíč	Typické vrácené hodnoty	Popis
<b>errors</b>	pole chybových objektů	<p>Chybový objekt obsahující pole všech chyb, které nastaly. Objekt <b>errors</b> je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče <b>code</b> je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři.</p> <p>Další zpřesnění důvodu chyby může být v hodnotě klíče <b>field</b> pro chybové kódy EDIR_FIELD_NAME_UNKNOWN a EDIR_FIELD_VALUE_ERROR, identifikující klíč nebo hodnotu nesplňující validační pravidla. V odpovědi se mohou vrátit následující chybové kódy:</p> <ul style="list-style-type: none"> <li>• EDIR_UUID_DOES_NOT_EXIST - záznam s daným uuid neexistuje (tj. nedá se aktualizovat).</li> <li>• EDIR_UUID_IS_MISSING - chybí povinný klíč <b>uuid</b>.</li> <li>• EDIR_UUID_INVALID_FORMAT - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány pro prázdný uuid.</li> <li>• EDIR_FIELD_NAME_UNKNOWN - neznámý klíč. Seznam dostupných klíčů pro dané zařízení najdete v kapitole / <b>api/dir/template</b>.</li> <li>• EDIR_FIELD_VALUE_ERROR - zadaná hodnota nesplňuje validační kritéria (hodnota není platná).</li> <li>• EDIRLIM_PHOTO - byl dosažen maximální počet fotografií v úložišti. Další fotografie se nedají přidávat.</li> <li>• EDIRLIM_FPT - byl dosažen maximální počet šablon otisků prstů v úložišti. Další otisky prstů se nedají přidávat.</li> <li>• EINCONSISTENT - v hodnotách klíčů <b>validFrom</b> a <b>validTo</b> je nesrovnalost (<b>validFrom</b> musí být menší než <b>validTo</b>).</li> </ul>

## Příklad odpovědi

```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid":
"01234567-89AB-CDEF-0123-456789ABCDEF", "timestamp": 39 }, { "uuid":
"76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ { "code":
"EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uuid": "01234567-89A-CDEF-0123-456789ABCDEF",
"errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] }, { "uuid": "01234567-89AB-
CDEF-0123-456789ABCDEF", "errors": [ { "code": "EDIR_FIELD_NAME_UNKNOWN", "field":
"albert" } ] }, { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "errors": [ {
"code": "EDIR_FIELD_VALUE_ERROR", "field": "access.pin" } ] } ] } }
```

První záznam v adresáři je úspěšně aktualizován, vrátí se jeho **uuid** a **timestamp** změny.

Druhý záznam neexistuje (není záznam s tímto **uuid**).

Třetí objekt má špatný formát **uuid**.

Ve čtvrtém objektu je uveden neznámý klíč **albert**.

V pátém objektu je zadána nesprávná hodnota PIN kódu.

#### 5.14.4 api dir delete

Funkce **/api/dir/delete** slouží k vymazávání pole záznamů v adresáři.

##### Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

##### Metody

- PUT

##### Požadavek

Požadavek obsahuje parametry ve formátu **application/json**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>owner</b>	Ano, pokud users je omitted	řetězec	-	Všechny záznamy v adresáři se zadaným vlastníkem budou vymazány.
<b>users</b>	Ano, pokud owner je omitted	pole JSON objektů definujících identifikátory uuid	-	Pole musí obsahovat alespoň jeden objekt s klíčem <b>uuid</b> , který definuje, které pole má být vymazáno.

### Příklad požadavku

```
URL: https://192.168.1.1/api/dir/delete JSON (owner specified) { "owner": "My2N" }
JSON (uuid specified) { "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF" }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654" }, { "uuid": "76543210-68FF-18-3210-FEDCBA987654" } ] }
```

Pokud není v adresáři žádný záznam se zadaným vlastníkem, vrátí se prázdné pole.

Pokud není v adresáři žádné pole se zadaným uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, bude vymazán. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Třetí uuid má nesprávný formát a vrátí se chyba.

### Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče **uuid** a **timestamp**.



- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče odpovědi JSON v poli **users**

Klíč	Typické vrácené hodnoty	Popis
<b>uid</b>	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) vymazaného nebo nezměněného záznamu.
<b>timestamp</b>	celé číslo	Časová značka změn provedených v adresáři. Více informací o použití časové značky najdete v kapitole <b>api/dir/query</b> . Časová značka je uvedena, jen když je změna v adresáři provedena úspěšně.
<b>errors</b>	pole chybových objektů	<p>Chybový objekt obsahující pole všech chyb, které nastaly. Objekt <b>errors</b> je uveden, jen když se změnu v adresáři nepodařilo provést.</p> <p>V hodnotě klíče <b>code</b> je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři.</p> <p>V odpovědi se mohou vrátit následující chybové kódy:</p> <ul style="list-style-type: none"> <li>• EDIR_UUID_DOES_NOT_EXIST - záznam se zadaným identifikátorem uuid neexistuje (tj. nedá se vymazat).</li> <li>• EDIR_UUID_INVALID_FORMAT - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány jako prázdný uuid.</li> </ul>

## Příklad odpovědi

```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uid": "01234567-89AB-CDEF-0123-456789ABCDEF", "timestamp": 39 }, { "uid": "76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ { "code": "EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uid": "76543210-68FF-18-3210-FEDCBA987654", "errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] } ] } }
```

První záznam v adresáři je úspěšně vymazán, vrátí se jeho **uid** a **timestamp** změny.

Druhý záznam neexistuje (neexistuje záznam s daným **uid**).

Třetí objekt má špatný formát **uuid**.

### 5.14.5 api dir get

Funkce **/api/dir/get** slouží k vyhledávání pole záznamů v adresáři a jejich zadaných polí.

#### Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

#### Metody

- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich objektovém zobrazení najdete v kapitole **api/dir/template**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Počet	Očekávaná hodnota	Výchozí hodnota	Popis
<b>fields</b>	Ne	pole řetězců	všechna pole s jinými hodnotami než výchozími	V odpovědi musí být specifikována jména požadovaných polí, pokud ne, vrátí se všechna pole s nevýchozími hodnotami, pokud se zadá prázdné pole, vrátí se všechna dostupná pole. Přehled všech dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole <b>api/dir/template</b> . Neznámé názvy polí budou ignorovány.
<b>users</b>	Ne	pole JSON objektů definujících uuid	-	Pole musí obsahovat alespoň jeden objekt s klíčem <b>uuid</b> , který definuje záznam, jehož pole se mají vrátit. Pokud klíč chybí nebo je zadáno prázdné pole, vrátí se prázdné pole.

## Příklad požadavku

```
URL: https://192.168.1.1/api/dir/get JSON { "fields": [ "name", "email",
"callPos.peer", "callPos[1].grouped" ], "users": [ { "uuid": "01234567-89AB-
CDEF-0123-456789ABCDEF" }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654" },
{ "uuid": "76543210-68FF-18-3210-FEDCBA987654" } ] }
```

Pokud není v adresáři žádné pole se zadaným uuid 01234567-89AB-CDEF-0123-456789ABCDEF, zařízení odpoví chybovým kódem (viz dále). To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654

Jestliže je v adresáři záznam s uuid 01234567-89AB-CDEF-0123-456789ABCDEF, jeho zadaná pole (v příkladu název, e-mail, telefonní čísla všech volajících stanic a pro druhou volající stanic i skupina) budou vrácena. To samé platí pro druhý uuid 76543210-68FF-18CA-3210-FEDCBA987654.

Identifikátor uuid 76543210-68FF-18-3210-FEDCBA987654 má nesprávný formát.

**Odpověď**

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Více informací o použití klíče **series** najdete v kapitole **api/dir/query**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

**Tip**

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Název klíče	Typické vrácené hodnoty	Popis
<b>uuid</b>	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) nalezeného záznamu.

Název klíče	Typické vrácené hodnoty	Popis
<b>Various keys</b>	různé	Zadaná pole záznamu v adresáři, která jsou vrácena. Viz <b>api/dir/template</b> .
<b>timestamp</b>	celé číslo;	Časová značka změn naposledy provedených pro každý vrácený záznam v adresáři. Více informací o použití časové značky najdete v kapitole <b>api/dir/query</b> . Časová značka je uvedena, jen když je záznam v adresáři vrácen.
<b>errors</b>	pole chybových objektů	<p>Chybový objekt obsahující pole všech chyb, které nastaly. Objekt <b>errors</b> je uveden, jen když se změnu v adresáři nepodařilo provést. V hodnotě klíče <b>code</b> je obsažen chybový kód, který ukazuje důvod neúspěšného provedení změny v adresáři. V odpovědi se mohou vrátit následující chybové kódy:</p> <ul style="list-style-type: none"> <li>• EDIR_UUID_DOES_NOT_EXIST - záznam se zadaným identifikátorem uuid neexistuje (tj. nedá se aktualizovat).</li> <li>• EDIR_UUID_INVALID_FORMAT - uuid nemá platný formát, který je "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX", kde X může být jakákoli hexadecimální číslice. Všechny nuly jsou rezervovány jako prázdný uuid.</li> </ul>

## Příklad odpovědi

```
{ "success": true, "result": { "series": "6423407687606431951", "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@abcd.cz", "callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ], "timestamp": 39 }, { "uuid": "76543210-68FF-18CA-3210-FEDCBA987654", "errors": [ { "code": "EDIR_UUID_DOES_NOT_EXIST" } ] }, { "uuid": "76543210-68FF-18-3210-FEDCBA987654", "errors": [ { "code": "EDIR_UUID_INVALID_FORMAT" } ] } ] } }
```

První záznam v adresáři je úspěšně vrácen, jeho **uuid** a **timestamp** jsou vráceny.

Druhý záznam neexistuje (neexistuje záznam s daným **uuid**).

Třetí objekt má špatný formát **uuid**.

### 5.14.6 api dir query

Funkce **/api/dir/query** slouží k vyhledávání pole záznamů v adresáři podle iterátoru časových značek a jejich zadaných polí.

#### Skupiny služeb a privilegií

- Služba je System.
- Privilegia jsou Systém – řízení.

#### Metody

- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu **application/json**. Více informací o jednotlivých parametrech záznamu v adresáři a jejich objektovém zobrazení najdete v kapitole **api/dir/template**.



Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>series</b>	Ne	řetězec	aktuální série zařízení	Řetězec představuje počet sérií časových značek v zařízení. Není-li klíč zadán, je uvažována současná série zařízení. Jestliže se zadaná série liší od současné série zařízení, zařízení vrátí svou sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.
<b>fields</b>	Ne	pole řetězců	všechna pole s jinými než výchozími hodnotami	V odpovědi musejí být specifikovány názvy požadovaných polí, pokud ne, vrátí se všechna pole s nevýchozími hodnotami, pokud se zadá prázdné pole, vrátí se všechna dostupná pole. Přehled všech dostupných klíčů v JSON definici záznamu v adresáři najdete v kapitole <b>api/dir/template</b> . Neznámé názvy polí budou ignorovány.
<b>iterator</b>	Ne	objekt JSON	{"timestamp": 0}	Klíč definuje iterátor dotazu (iterátor časové značky je podporován). Iterátor časové značky je celočíselná časová značka, která indikuje, že se má vrátit první záznam v adresáři (pomocí funkcí <b>api/dir/create</b> , <b>api/dir/update</b> nebo <b>api/dir/delete</b> se vždy vrací poslední časová značka). Je-li hodnota iterátoru časové značky nulová, vrátí se všechny záznamy v adresáři. Je-li vyšší než aktuální hodnota časové značky v zařízení, zařízení vrátí svou sérii, nejvyšší hodnotu časové značky.

## Příklad požadavku

```
URL: https://192.168.1.1/api/dir/query JSON { "series": "2229480630597592840",  
"fields": [ "name", "email", "callPos.peer", "callPos[1].grouped" ], "iterator":  
{ "timestamp": 6 } }
```

Nesouhlasí-li klíč **series** s aktuálním klíčem **series** v zařízení, zařízení vrátí svou sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.

Je-li zadaná časová značka nižší než aktuální nejvyšší časová značka, vrátí se všechny vyšší časové značky.

Zařízení je schopno zvládnout až 10000 unikátních uživatelských identifikátorů. Když se tento počet zvýší, zařízení vrátí klíč **invalid**, který znamená, že adresář má neznámou historii (některé záznamy byly vymazány a nejsou již uloženy v zařízení).

Je-li zadaná časová značka nižší než neplatná časová značka, zařízení vrátí svou aktuální sérii, nejvyšší hodnotu časové značky a neplatnou časovou značku.

## Odpověď

Odpověď je ve formátu **application/json**. Objekt **result** obsahuje klíče **series** a **users**.

Klíč **users** obsahuje pole objektů, které obsahují klíče a hodnoty z výsledku požadavku (viz následující tabulka).

### ✓ Tip

- Více o struktuře JSON odpovědi se dozvíte z příkladu na konci této kapitoly.

Tabulka 2. Klíče JSON odpovědi v poli **users**

Název v klíče	Typické vrácené hodnoty	Popis
<b>uuid</b>	uuid	Unikátní uživatelský identifikátor (Unique User Identifier) nalezeného záznamu.
<b>Various keys</b>	různé	Zadaná pole záznamu v adresáři, která jsou vrácena. Viz <b>api/dir/template</b> .
<b>timestamp</b>	celé číslo	Časová značka změn naposledy provedených pro každý vrácený záznam v adresáři. Časová značka je uvedena, jen když je záznam v adresáři vrácen.

## Příklad odpovědi

```
{ "success": true, "result": { "series": "2229480630597592840", "users": [ { "uuid": "01234567-89AB-CDEF-0123-456789ABCDEF", "name": "ABCD", "email": "abcd@abcd.cz", "callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ], "timestamp": 7 }, { "uuid": "A6543210-68FF-18CA-3210-FEDCBA987654", "name": "DEFG", "email": "defgd@defg.cz", "callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ], "timestamp": 9 }, { "uuid": "044197A7-54AD-7577-6EEA-787A6097263E", "name": "HIJK", "email": "hijk@hijk.cz", "callPos": [ { "peer": "" }, { "peer": "", "grouped": "false" }, { "peer": "" } ], "timestamp": 10 } ] } }
```

Vrátí se tři záznamy v adresáři, které mají časovou značku vyšší než 6 (v tomto případě je maximální časová značka v adresáři 10).

## 5.15 api mobilekey

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/mobilekey**.

- [5.15.1 api mobilekey config](#)

### 5.15.1 api mobilekey config

Funkce **api/mobilekey/config** slouží ke čtení a zapisování identifikátorů lokace a šifrovacích klíčů pro autentizaci přes Bluetooth.

#### Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (sledování).

#### Metody

- **GET** – čtení identifikátorů lokace a šifrovacích klíčů
- **PUT** – zapisování identifikátorů lokace a šifrovacích klíčů

#### Požadavek

Pro metodu **GET** se nepoužívají žádné parametry.

Parametry požadavku **PUT** se ukládají ve formátu **application/json**.

Tabulka1. Klíče JSON požadavku PUT

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>location</b>	Ne	Řetězec o maximální délce 127 znaků	–	<b>location</b> definuje lokaci zařízení pro účely autentizace přes Bluetooth. Akceptován je každý řetězec, který danou lokaci jednoznačně definuje. Lokace je pak přenášena zařízeními 2N a slouží Bluetoothovému autentizačnímu zařízení k vybrání relevantních autentizačních parametrů.

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>keys</b>	Ne	Pole objektů obsahující šifrovací klíče	–	<p><b>keys</b> obsahuje šifrovací klíče k zabezpečení komunikace mezi zařízením 2N a zařízením určeným k provedení autentizace přes Bluetooth. Objekty v poli mají tyto klíče:</p> <ul style="list-style-type: none"> <li>• <b>type</b> – algoritmus, v současnosti je podporován algoritmus RSA, tento klíč je nepovinný,</li> <li>• <b>key</b> – data šifrovacího klíče (formát DER kódovaný v Base64), používají se 1024bitové šifrovací klíče, tento klíč je povinný,</li> <li>• <b>ctime</b> – čas vytvoření ve tvaru Unixového času, 32bitové celé číslo bez znaménka, tento klíč není povinný.</li> </ul>

## Příklad požadavku PUT

```

URL: https://192.168.1.1/api/mobilekey/config JSON: { "location": "LocationUniqueID",
"keys": [ { "type": "rsa", "key":
"MIICXAIBAAKBgQCXmIX1U7wGFW3FiDdhq7BIktIc4lg7X2IMxLE83I75S3BRPL/
7LCAefnMUJL0uyyFdeMpRo0VhVs/iPfnYPNf4AiQ04LIQh8tSKDeat5IfXSMY9zXMYHeB0Bg19R+/
uShyJsnLoJoB5MJDowwkOuSMIskK+dA17+3E/
Y+ujhhpCQIDAQABAoGAfzHOVAUp4cDhFbgxH5Y6lun5uZqAhXCGiEgQngxB0hJ97uuV+V0QpgVa8S/
SPAzbtd2/g7YIQB/
i10VDWJfUbeiuBhr6ZHwk5jfcFF0KkmTQtEBd4bbCz+Fwyoy19DUXdsLNMf8GW4eWhooX+NCqc2sfl04Nz+S
pXmqpsMIc/ECQQDk63xVnRqPCgG3fqPLVGWkQL9wmYAIUP8MrdoAFRYfSC/
LrjX55lCRj4mAnSzRQfNclSEz2mITkoaCcjnl1TmXAKEAqYdjMEhIrg4LpYzqZDOF6v6w/
sUcLkepktTBYCFFV+YgOrlPr7akR8XtED8X/6QHwWciphp/50BoJ/
KRAWGxWJAZ0YNe5o6pxk+mQed0AotKK0A5w15A0d3KMMqzaag2k/
4sAzR8QGEi4aT4+AEngsAvV3R8tCsum06JxNdLnu51QJA06abzBljFxtztajDwMYwV00R09P3eoFuYmPEVgj
oi4jIQabd2R4oZiPNaw9sYHyCKdVlcS1Q7+CZqv/
QdKLWQJBAKeoGxqcpDHvMtzgcSj3lZz0Z8dWmgtTF7Q05boHhxtZ1SEo3MvlicVue8U1tV2XjUR6r7YueusM
9GxbBqr5YI=", "ctime": 1608047606 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCfyMHsTjPKf3Dv00gWmRAR5UZrpt3tBy3kBvPv4R4o9H7Zzse7+yKwfPTddKJQ0L1IrCX
06Zo8SZAMotjjpMy1M9K27ZB95YtAYiGLLRWeLAJUKL4gixgkHeS+T8uQxLW7/
etqwU00uPmd94ZEZY226ChdKQW3zge2WEtuQ5oCwIDAQABAoGAZCp6RyUPGpahuFZ9fpmKddJqCduH4paqmfh
hNu8coHQyIqQoT9CgPKwxqhJmlVxz6rCAe+1WmNrZ27LT5uluJKViu0XnLV7FHG2smagjQ3rPepg0GcayphuI
lHikaBCafxnCRV/
E1Ifg08d1xK4cK858yMjpoEgDdEJi0R2qmECQQDXqtWGiXYSRnZzR90eCjrip6IIQqJuARE91L0Ly0hkPzCiP
Pf2IrT1JQsw6Tu0ZTm3NjzZ0VSEdZU6s2NcKhsnAkEAvap5GacBi9EZ9lsiaQj/
dVA6LbUnBCo7qwRj7SUyW6ikCvmLjdpjR80twj3FTAXB0sTeWgyT42HmBpPX2dKfQJBAMc5Ml9nhAaFyM3dS
MmDMbpGmEuBIoLzWXYkvNB+EsChG6aw4SnsVnx6LCYY2rVR2eR1oLv+F8UL3I2XEa5rmkCQBZXhnxnF9+Iei
5y/dKxpKYFFVvdCYOMFgtHMR42SHyD2Q8R6Dvpex2Ml4EYJULxr0TEqz6Z75M/
cMGSF9d9K2ECQQDEffsJoyjYwY2rGbPX8N5d9yrp3HLRbH4RjFGR0zCbSaA+PTQwxu2q1Asd8g7LN95Umyvli
ddJgayDIwnJSGse", "ctime": 1608044538 } ] }

```

Zařízení 2N umožňují použití maximálně 4 šifrovacích klíčů najednou. První šifrovací klíč v poli se považuje za primární šifrovací klíč a ostatní klíče jsou sekundární. Jestliže se Bluetoothové zařízení autentizuje jakýmkoli sekundárním šifrovacím klíčem, zařízení 2N je vyzve k nahrazení tohoto šifrovacího klíče klíčem primárním. Proto by se vždy měl na začátek pole přidávat nejnovější šifrovací klíč.

Je-li předloženo pole kratší než 4, jsou chybějící šifrovací klíče vymazány (nahrazeny prázdným objektem).

Klíč **type** není povinný. Je-li vynechán algoritmus, zařízení 2N automaticky předpokládá použití algoritmu RSA (**rsa**).

Parametr klíče **ctime** není povinný. Je-li čas vytvoření vynechán nebo neplatný, zařízení 2N zobrazí v konfiguračním webu datum 1. ledna 1970 00:00:00 a pro tento šifrovací klíč nevrátí **ctime**.

## Odpověď

Odpověď na požadavek **GET** je ve formátu **application/json**. **Výsledný objekt** obsahuje klíče **location** a **keys**.

Odpověď na požadavek **PUT** neobsahuje žádné podrobnosti. Např. když je uvedena neplatná hodnota šifrovacího klíče, klíč nebude zapsán bez notifikace.

Tabulka 2. Odpověď na klíče JSON požadavku GET

Klíč	Typické vrácené hodnoty	Popis
<b>location</b>	Řetězec	Identifikátor lokace zařízení 2N. Podrobnosti najdete v sekci Požadavek.
<b>keys</b>	Pole objektů obsahující šifrovací klíče	Délka pole je vždy 4 (za chybějící klíče se vrací prázdné objekty). Podrobnosti a struktura objektů v poli jsou popsány v sekci Požadavek.

## Příklad odpovědi na požadavek GET

```
{ "success": true, "result": { "location": "54-1046-0745", "keys": [ { "type": "rsa",
"key": "MIICXAIBAAKBgQCXmIX1U7wGFW3FiDdhq7BIktIc4lg7X2IMxLE83I75S3BRPL/
7LCAefnMUJL0uyyFdeMpRo0VhVs/iPfnYPNf4AiQ04LIQh8tSKDeat5IfXSMY9zXMyHeB0Bg19R+/
uShyJsnLoJoB5MJDowkOuSMIskK+dA17+3E/
Y+ujhhpCQIDAQABAoGAfzHOVAUp4cDhFbgxH5Y6lun5uZqAhXCGiEgQngxB0hJ97uuV+V0QpgVa8S/
SPAzbtd2/g7YIQB/
i10VDWJfUbeIuBhr6ZHwk5jfcff0KkmTQtEBd4bbCz+Fwyoy19DUXdsLNMf8GW4eWhooX+NCqc2sflo04Nz+S
pXmqpsMIc/ECQQDk63xVnRqPCG3fqPLVGWkQL9wmYAIUP8MrdoAfrYfSC/
LrjX55lCRj4mAnSzRQfNclSEz2mITkoaCcjnl1TmXAKEAqYdjMEhIrg4LpYzqZDOF6v6w/
sUcLkepktTBYCFFV+YgOrlPr7akR8XtED8X/6QHwWciphp/50BoJ/
KRAWGxWJAZ0YNe5o6pxk+mQed0AotKKOA5w15A0d3KMMqzaag2k/
4sAzR8QGEi4aT4+AEngsAvV3R8tCsum06JxNdLnu51QJA06abzB1jFXtztajDwMYwV00R09P3eoFUtYmPEVgj
oi4jIQabd2R4oZiPNaw9sYHyCKdVlcS1Q7+CZqv/
QdKLWQJBAKeoGxqcpDhVmtzgcSj3lZz0Z8dWmgtTF7Q05boHhxtZ1SEo3MvlicVue8U1tV2XjUR6r7YueusM
9GxbBqr5YI=", "ctime": 1608047754 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCfyMHsTjPKf3Dv00gwMrQAR5UZrpt3tBy3kBvPv4R4o9H7Zzse7+yKwFPTddKJQ0L1IrCX
06Zo8SZAMotjjpMy1M9K27ZB95YtAYiGLLRWeLAJUKL4gixgkHeS+T8uQxLW7/
etqwU00uPmd94ZEZy226CHdKQW3zge2WEtuQ5oCwIDAQABAoGAZCp6RyUPGpahuFz9fpmKddJqCduH4paqmfh
hNu8coHqYIqQoT9CgPKwxqhJmlVxz6rCAe+1WmNrz27LT5uluJKViU0XnLV7FHG2smagjQ3rPepg0Gcayphui
IlHikaBCafxnCRV/
E1Ifg08d1xK4cK858yMjpoEgDdEJi0R2qmECQQDXqtWGiXYSRnZzR90eCjrip6IIQqJuARE91L0Ly0hkPzCiP
Pf2IrT1JQsw6Tu0ZTm3NjzZ0VSEdZU6s2NcKHsNAkEAvap5GacBi9EZ9lsiaQj/
dVA6LbUnBCo7qwRj7SUyW6ikCvmcljdpjR80twj3FTAXB0sTeWgyT42HmBpPX2dKfQJBAMc5Ml9nhAaFyM3dS
MmDMbpGmEuBIoLzWXYkvNB+EsChG6aW4SnsVnx6lCYy2rVR2eR1oLv+F8UL3I2XEa5rmkCQBZXhnxnF9+Iei
5y/dKxpKYFFVvdCYOMFgtHMR42SHyD2Q8R6Dvpex2Ml4EYJULxR0TEqz6Z75M/
cMGSF9d9K2ECQQDEffSJoyjYwY2rGbPX8N5d9yrp3HLRbH4RjFGR0zCbSaA+PTQwxu2q1Asd8g7LN95Umyvli
ddJgayDIwnJSGse", "ctime": 1608046389 }, { "type": "rsa", "key":
"MIICXQIBAAKBgQCwXVv2CNCUFgoQBQ5NjaLJVEWuAFryK/
h9jfNe+qDuFfS+itWsfvvyvMkUhhiiDpCpgo0gqEipkYa0q3maPKPS4CJXZBFo++JSzsgw6a/
VxH0n8joHfJf6nIEcCGcuMAa/
HOEOZq7uL7n2jTsyVnnDbYCLXENh4Np9izSX23QIDAQABAoGBAI5iDFDMrfAw5p0dpqWpv/
SXnoUsIkg0mYeu9ULzU0grVLKVKW22Jm30elyWyKwIUaid0zBXfHp7NRTk09V1dSnS5Cnu073tye9MV5TeLqj
MSBVCSPZWK//
hu1VaRAL9UTZc+1e277l0B8c1Fup4uxR4b757brrcLNkjt1U4Hh5AkEA4mFz+IrgTtdiLNLQdww5B3ZELma0l
+lkyGc50hvqy2TDNGGiKGpQymEd/4ySHBmaGnoh9ZFxnC/
ItrNEXBGdawJBAMds0d2qDdb0Sie2TpsGJs5eEUrLX6yW/
w+si04SXczCTnJXckZyj79eE0cNRTRK+SuDsN+8+wm03b9CZqx0xtcCQQCUkukOAfddRzaDvIhc2YTERPZSjb
SgNuL0+LL8Fp5uht8mjb1jTNATaTHK+nMaRiNBpU6MYLxziVjtr5H56wWpAka0fXYVtEcEPTQjk8bI4yufs7
XMwSxxuTH2WAJWeg6lwJS8lvv2Y0gmT/VuAnM89b17ynFGLbxQxt21iF0RR/
tAkA0nJmkbD3daWYAudtOQjzema30009r90NZ/
Khj5tQpv8gLe6EEpyFUNQnodNoTmkIJJmPLlBjyx+zTspdE+C" }, { } ] } }
```

Výchozí hodnota parametru **location** je sériové číslo zařízení 2N. Chcete-li přidat do jedné lokace více zařízení, patřičně tuto hodnotu změňte.

## 5.16 api lpr

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/lpr**.



- [5.16.1 api lpr licenseplate](#)
- [5.16.2 api lpr image](#)

### 5.16.1 api lpr licenseplate

Funkce **api/lpr/licenseplate** slouží k řízení přístupu pomocí rozpoznávání registračních značek vozidel (SPZ).

#### Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (řízení).

#### Metody

- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu **application/json**.

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
lprUuid	Ano	uuid řetězec	–	Identifikátor uuid události rozpoznání registračních značek generovaný systémem Rozpoznávání registračních značek.
lprID	Ano	ID (číslo)	–	Interně určený identifikátor označující jednu registrační značku v rámci jednoho zaznamenaného příjezdu vozidla. V rámci jednoho příjezdu může být tatáž registrační značka rozpoznávána vícekrát. V případě, že kamera získá z dalšího rozpoznání přesnější data, upřesní se plateText, zatímco lprID zůstává stejné. Každá jednotlivá událost rozpoznání generuje vlastní lprUuid.

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
accessPoint	Ano	0 nebo 1	–	Signalizuje, že vozidlo s detekovanou registrační značkou vjíždí (0) do objektu nebo z něj vyjíždí (1) – podle toho se pro událost v zařízení 2N použijí nastavená přístupová pravidla.
plateText	Ano	řetězec registrační značky	–	Text rozpoznané registrační značky, který slouží k identifikaci uživatele v seznamu uloženém v zařízení 2N.
lprDir	Ne	0 až 3	–	Určuje detekovaný směr pohybu vozidla/registrační značky dle nastavení směru na kameře.  Možné směry: <ul style="list-style-type: none"> <li>• 0 = Unknown</li> <li>• 1 = Undefined (In nebo Out)</li> <li>• 2 = In</li> <li>• 3 = Out</li> </ul>
plateImage	Ne	obrázek šifrovaný pomocí base64	Bez obrázku	Obrázek, na němž byla rozpoznána registrační značka. Maximální velikost obrázku je 256 kB.

### Příklad požadavku

URL: <https://192.168.1.1/api/lpr/licenseplate>

JSON:

```
{
  "lprUuid": "bc4baad9-d2cd-4706-986f-b942963bfa9f",
  "lprID": 143289,
  "accessPoint": 0,
  "plateText": "ABC123456",
  "plateImage": "/9j/4AAQSkZJRgABAQEASABIAAD//gATQ3JlYXRlZCB3aXRoIEedJTVD/
4gKwSUNDX1BST0ZJTEUAAQEAAAKgbGNTcWQwAABtbnRyUkdCIFhZWiAH5AAMABIAcGAAHAA5hY3NwTVNGVAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA9tYAAQAAAADTLWxjbXMAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA1kZXNjAAABIAAAAEbjcHJ0AAAABYAAAADZ3dHB0AAAABmAAAABRja
GFkAAABrAAAACxyWFlaAAAB2AAAABRiWFlaAAAB7AAAABRnWFlaAAACAAAAABRYVFJDAAACFAAAACBnVFJD
AAACFAAAACBivVFJDAAACFAAAACBjaHJtAAACNAAAACRkbW5kAAACWAAAACRkbWRkAAACfAAAACRtbHVjAAA
AAAAAAAAEAAAAMZW5UuAAACQAAACAEcASQBNAAFAAIABiAHUAAaQBSAHQALQBpAG4AIABzAFIARwBCBwX1Yw
```





```
gQxORiQcAULChPSAnBIjfZK/jq/  
n6kgwAoGEk3GyUe9dkpQGETcssrR6KISwQ2FZyt8EYTYon6GmNq0qZEaLkx1TSu7GXkau+DBJ56GgmuFQBA  
K2wb0Gi4QAAADAYND7Rcc+haSit9fMbzVZhtiurHTx0lcwIX0YYjgOKRJDRLxW61mZB0Zfaj4rRJ56GhH  
w/gbACs3wbTNEntl+hZmfu1Q+6En7E9CoIEx4+gAo4HsgPF8P3FS0cg5JkphItLB5NF3talYIF7YDt/  
vzRjaqHY9+9GR+DEnnoaB/  
VgRJ1+1XuoKZAASIIAMzLNHPYwFDwu+yKFxWCL4bDo+C72tSoAM8q0Rkd69iV/  
et3wGJPPQ0AVC+OQH2VALyIviwDfMMyi0/  
BQbv25MCRwi2Infrd7GqYDYMDAnx6VM6aFCqgdmHNEQRAApKMhKCTEULN5LyMdqGrxgLwXMnZ+nROGk4Sra  
SZUFElukB8DBh4kHcnG3o3HEs80g20tA0SmwTqpYMLE4EMW2o/  
4k4UYR2aHLyzhYSR5IJ4NEgtSnXdIiACDAyErpZAE8WZKDr/AEqKgE4iMnhFI2W9LL4JIH1fMdw/  
LYQIX4dIuVCg674kgSWwA+pyumSPjX0J+UGDBgwYMGDBgwYMiFYNegNP/9k="
```

### **Odpověď**

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>success</b>	true, false	Hodnota je true, když je požadavek úspěšně zpracován. Dojde-li k chybě, hodnota je false a v popisu chyby error jsou k dispozici další informace.

## Příklad odpovědi

```
{ "success": true }
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 13 (příliš velká data parametru) znamená, že odpověď nebyla zpracována a je nutné zaslat požadavek znovu s menším obrázkem nebo bez obrázku.

Následně obdržené duplicitní platné požadavky jsou ignorovány (v paměti se uloží jen posledních deset úspěšných požadavků). Je možné pokusit se poslat požadavek znovu, když nepřijde od zařízení 2N žádná odpověď, bez riskování otevření duplicitní bariéry nebo duplicitního uložení události.

## 5.16.2 api/lpr/image

Funkce **api/lpr/image** umožňuje získávat obrázky ze systému rozpoznávání registračních značek (SPZ).

**Skupiny služeb a privilegií**

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu (sledování).

**Metody**

- GET
- POST

**Požadavek**

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>plateText</b>	Ano	Řetězec s textem registrační značky	-	Text rozpoznané registrační značky, který slouží k identifikaci obrázku, který má zaslán (dá se uložit až pět obrázků). Patří-li ke stejnému textu registrační značky dva nebo více obrázků, vrátí se ten nejnovější.

### Příklad požadavku

URL: <https://192.168.1.1/api/lpr/image?plateText=ABC123456>

### Odpověď

Úspěšná odpověď je ve formátu **image/jpeg**.

Může dojít k různým chybám (např. chybějící povinný parametr). Chyby se vracejí ve formátu json s odpovědním kódem 200. Není-li k předloženému textu registrační značky přiřazen žádný obrázek, vrátí se chyba 15 - "data nejsou k dispozici".

## 5.17 api accesspoint

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/accesspoint**.

- [5.17.1 api accesspoint blocking ctrl](#)
- [5.17.2 api accesspoint blocking status](#)
- [5.17.3 api accesspoint grantaccess](#)

### 5.17.1 api accesspoint blocking ctrl

Funkce **api/accesspoint/blocking/ctrl** slouží ke kontrole blokování přístupu pro jednotlivé přístupové body.

#### Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

#### Metody

- GET
- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry URL požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
id	Ano	Celé číslo (0, 1)	-	Určuje identifikátor přístupového bodu, který se má kontrolovat (0 pro Příchod a 1 pro Odchod).
action	Ano	Řetězec (on, off)	-	Určuje, zda má být blokování daného přístupového bodu zapnuto / vypnuto.

#### Příklad požadavku

URL: `https://192.168.1.1/api/accesspoint/blocking/ctrl?id=0&action=on`



**Odpověď**

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>success</b>	true, false	Hodnota je true, když je požadavek úspěšně zpracován (tj. blokování přístupu je v požadovaném stavu bez ohledu na nastalou změnu).

**Příklad odpovědi**

```
{ "success": true }
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 18 (přístupový bod nepovolen) znamená, že požadavek nebyl zpracován, protože daný přístupový bod byl v té době nepovolen.

**5.17.2 api accesspoint blocking status**

Funkce **api/accesspoint/blocking/status** vrací stav blokování přístupu pro jednotlivé přístupové body.

**Skupiny služeb a privilegií**

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – sledování.

**Metody**

- GET
- POST

**Požadavek**

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry požadavku URL

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
id	Ne	Celé číslo (0, 1)	Všechny	Určuje, pro který přístupový bod se má vrátit stav blokování přístupu. Jestliže je tento parametr vynechán, stav blokování přístupu se vrátí pro všechny přístupové body.

### Příklad požadavku

URL: <https://192.168.1.1/api/accesspoint/blocking/status?id=0>

### Odpověď

Odpověď je ve formátu **application/json**. Hodnota klíče `result` obsahuje jeden klíč, `accessPoints`, který obsahuje pole s objektem pro každý přístupový bod (pole má délku 1, jestliže byl v požadavku přístupový bod specifikován). Objekty v poli obsahují následující klíče.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>id</b>	Celé číslo (0, 1)	Identifikuje přístupový bod (0 pro příchod, 1 pro odchod).
<b>blocked</b>	Boolean (true, false)	Obsahuje aktuální stav blokování přístupového bodu (true, když je přístupový bod blokován, false, když není blokován).

### Příklad odpovědi

```
{ "success": true, "result": { "accessPoints": [ { "id": 0, "blocked": true }, { "id": 1, "blocked": false } ] } }
```

Může dojít k různým chybám (např. nedostatečná privilegia).

### 5.17.3 api accesspoint/grantaccess

Funkce **api/accesspoint/grantaccess** slouží k povolení vzdáleného přístupu příslušnému uživateli (konkrétního zaměstnance/uživatele nebo uživatele zahrnutého do obecné skupiny, např. návštěvník). Povolení vzdáleného přístupu lze udělit také pod přihlášeným účtem, kdy je ze záznamu zřejmé, který uživatel přístup umožnil.

#### Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

#### Metody

- GET
- POST

#### Požadavek

Požadavek obsahuje parametry ve formátu URL.

Tabulka 1. Parametry URL požadavku

Název parametru	Povinný	Očekávané hodnoty	Výchozí hodnota	Popis
<b>id</b>	Ano	Celé číslo (0, 1)	–	Určuje identifikátor přístupového bodu, který se má kontrolovat (0 pro Příchod a 1 pro Odchod).
<b>user</b>	Ano	uuid	–	Identifikace uživatele, jehož jménem má být otevřeno (a jehož přístupová nastavení mají být vzata v potaz).

#### Odpověď

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
<b>success</b>	true, false	Hodnota je true, když je požadavek úspěšně zpracován (tj. blokování přístupu je v požadovaném stavu bez ohledu na nastalou změnu).

Klíč	Typické vrácené hodnoty	Popis
<b>reason</b>	invalidAp , invalidCredential, accessBlocked	Klíč se zobrazuje v případě odpovědi accessGranted:false. V případě úspěšného přístupu není tento klíč zobrazen.

### Příklad odpovědi

```
{
  "success" : true,
  "result" : {
    "accessGranted" : false,
    "reason" : "invalidAp"
  }
}
```

Může dojít k různým chybám (např. chybějící povinný parametr). Chybový kód 18 (přístupový bod nepovolen) znamená, že požadavek nebyl zpracován, protože daný přístupový bod nebyl v té době povolen.

## 5.18 api lift

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/lift**.

- [5.18.1 api lift grantaccess](#)

### 5.18.1 api lift grantaccess

Funkce api/lift/grantaccess slouží k aktivaci pater výtahu na základě autorizace v jiném zařízení.

#### Skupiny služeb a privilegií

- Skupina služeb je Access Control API.
- Skupina privilegií je Správa přístupu – řízení.

**Metody**

- GET
- POST

**Požadavek**

Požadavek obsahuje parametry ve formátu URL (nebo ve formátu application/x-www-form-urlencoded při použití metody POST).

Tabulka 1. Klíče požadavku JSON

Název klíče	Povinný	Očekávané hodnoty	Výchozí hodnoty	Popis
uuid	Ano	uuid	-	Uuid uživatele, kterému má být udělen přístup do jeho pater (podle nastavení Přístupových práv).
duration	NE	1 .. 600	doba trvání nastavená v cílovém zařízení	Definuje dobu aktivace pater. Je-li tento parametr vynechán, použije se výchozí hodnota uvedená v parametru <b>Switch-On Duration</b> .

**Příklad požadavku**

URL:

```
https://192.168.1.1/api/lift/grantaccess?user=09ebfd7d-24e4-4d58-ad02-804ad69938a6&duration=180
```

**Odpověď**

Odpověď je ve formátu **application/json**.

Tabulka 2. Klíče odpovědi JSON

Klíč	Typické vrácené hodnoty	Popis
success	true, false	Když je požadavek úspěšně zpracován, je hodnota true. Dojde-li k chybě, je hodnota false a v klíči error jsou další informace.

## Příklad odpovědi

```
{
  "success": true
}
```

Může dojít k různým chybám (např. chybějící povinný parametr). Vrátili-li se chybový kód 12, param: user (Uživatel nenalezen), požadavek nebyl zpracován, protože v adresáři cílového zařízení nebyl tento identifikátor uuid nalezen. Pokud zadaný uuid chybí nebo má nesprávný formát, zařízení odpoví chybovým kódem 11 (chybějící povinný parametr) nebo chybovým kódem 12 (neplatná hodnota parametru) v tomto pořadí.

Když dojde k aktivaci pater, která jsou již aktivní, bude použita delší doba trvání (zbývající čas versus nově požadovaná doba trvání).

Tento koncový bod API se dá použít k ovládní výtahových pater spolu se standardním řízením přístupu z jiného zařízení (např. posílat aktivační požadavek pro výtahová patra z interkomu na přístupovou jednotku, která je ve výtahu a komunikuje přímo s reléovými deskami).

## 5.19 api automation

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/automation**.

- [5.19.1 api automation trigger](#)

### 5.19.1 api automation trigger

Funkce **/api/automation/trigger** slouží pro aktivaci funkce automatizace události HttpTrigger.

Funkce je součástí služby **Automation API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Přístup k automatizaci**.

Pro tuto funkci lze použít metody **GET**.

Funkce je nastavena dle triggerId s parametry požadavku.

Odpověď je ve formátu **application/json** a obsahuje souhrn informací o zařízení:

**Příklad:**

```
{
  "success" : true
}
```

## 5.20 api cert

V podkapitolách jsou detailně popsány jednotlivé HTTP funkce dostupné pro službu **api/cert**.

- [5.20.1 api cert ca](#)
- [5.20.2 api cert user](#)

## 5.20.1 api cert ca

Funkce **/api/phone/config** slouží ke správě CA certifikátů.

Funkce je součástí služby **Systém API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém – řízení**.

U této funkce lze použít metodu **GET**, **PUT** nebo **DELETE**. Metoda **PUT** vrací informace o certifikátech v zařízení. Metoda **PUT** slouží k nahrání certifikátů do zařízení. Metoda **DELETE** slouží ke smazání certifikátu ze zařízení.

### Metoda GET

Parametry požadavku **GET**:

Parame tr	Popis
<b>id</b>	Volitelný parametr (string) identifikující CA certifikát. Hodnota parametru <b>id</b> je přednastavené id nebo otisk (hash) certifikátu. Pokud není <b>id</b> zadáno, odpověď obsahuje úplný seznam všech CA certifikátů v zařízení.

Odpověď je ve formátu **application/json** a může obsahovat následující parametry:

Para met r	Popis
<b>fing erp rint</b>	Otisk (hash) certifikátu
<b>sub ject ,iss uer</b>	Informace o subjektu nebo vydavateli: Common Name (CN), Organization (O), Organization Unit (OU), Location (L), State (S), Country (C)
<b>id</b>	Hodnota ( <i>řetězec</i> ) dříve nastavené identifikace certifikátu.
<b>sta rtD ate</b>	Datum začátku platnosti certifikátu

Parameter	Popis
<b>endDate</b>	Datum konce platnosti certifikátu
<b>protected</b>	Hodnota určující, že certifikát nemůže být smazán ( <code>true/false</code> ). Interní certifikáty s <b>id</b> začínajícím znakem # nemohou být smazány.
<b>systemOnly</b>	Hodnota určující, zda může daný certifikát uživatel zvolit jako certifikát pro jakoukoli službu. Pokud je hodnota <code>true</code> , certifikát se v seznamu pro výběr nezobrazí.

## Příklad 1: Získání dat všech certifikátů

```

GET /api/cert/ca //požadavek
{ //odpověď
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "4deea7060d80babf1643b4e0f0104c82995075b7",
        "subject" : {
          "CN" : "Thawte RSA CA 2018",
          "O" : "DigiCert Inc",
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "issuer" : {
          "CN" : "DigiCert Global Root CA",
          "O" : "DigiCert Inc",
          "OU" : "www.digicert.com",
          "C" : "US"
        },
        "startDate" : "2017-11-06T12:23:52Z",
        "endDate" : "2027-11-06T12:23:52Z",
        "allowRemove" : true
      },
      {
        "fingerprint" : "a8985d3a65e5e5c4b2d7d66d40c6dd2fb19c5436",
        "subject" : {
          "CN" : "DigiCert Global Root CA",
          "O" : "DigiCert Inc",

```



```

    "OU" : "www.digicert.com",
    "C" : "US"
  },
  "issuer" : {
    "CN" : "DigiCert Global Root CA",
    "O" : "DigiCert Inc",
    "OU" : "www.digicert.com",
    "C" : "US"
  },
  "startDate" : "2006-11-10T00:00:00Z",
  "endDate" : "2031-11-10T00:00:00Z",
  "protected" : false,
  "id" : "#my2n-utility",
  "systemUseOnly" : true
}
]
}
}

```

### Příklad 2: Získání informací pro certifikát zadaného **id**

```

GET /api/cert/ca?id=#my2n-utility //požadavek
{ //odpověď
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "a8985d3a65e5e5c4b2d7d66d40c6dd2fb19c5436",
        ...
        "id" : "#my2n-utility",
        ...
      }
    ]
  }
}
}

```

## Metoda PUT

Při opakovaném nahrání stejného certifikátu se původní certifikát přepíše. Je možné nahrát více certifikátů v jednom souboru ve formátu PEM. Soubor může obsahovat libovolné bloky, zpracovávají se pouze certifikáty. Pokud se některý z obsažených certifikátů nepodaří nahrát, žádný se neuloží a vrátí se kód chyby.

Parametry požadavku **PUT**:

Parametr	Popis
<b>blob-cert</b>	Povinný parametr obsahující soubor s certifikátem ve formátu DER nebo PEM.

Parametr	Popis
<b>id</b>	<p>Volitelný parametr (string) identifikující CA certifikát.</p> <p>Pokud je nahrán certifikát pod stejným <b>id</b>, původní certifikát s tímto <b>id</b> se přepíše.</p> <p>Při nahrávání souboru s více certifikáty se <b>id</b> nezadává.</p>

Odpověď je ve formátu **application/json** a obsahuje otisk nahraného certifikátu

Parametr	Popis
<b>fingerprint</b>	Otisk (hash) nahraného certifikátu
<b>replaced</b>	Otisk přepsaného certifikátu

Příklad:

```

PUT /api/cert/ca                                     //požadavek
{
  "success" : true,                                 //odpověď
  "result" : {
    "certificates" : [
      {
        "fingerprint": "9623fa35e414aa930ed22348a22d04a4c4fda26b"
      },
      {
        "fingerprint": "9623fa35e414aa930ed22348a22d04a4c4fda26b"
        "replaced": "9623fa26e414aa930ed22348a22d04a4c4fda26c"
      }
    ]
  }
}
-----
{
  "success" : false,                                //odpověď
  "error" : {
    "code" : 12,
    "param" : "blob-cert",
    "description" : "invalid certificate",
    "data" : "invalid_cert"
  }
}

```

## Metoda DELETE

Parametry požadavku **DELETE**:

Parameter	Popis
<b>id</b>	Povinný parametr (string) identifikující CA certifikát. Parametr <b>id</b> je přednastavené id nebo otisk (hash) certifikátu.

Odpověď je ve formátu **application/json**.

Příklad:

```
DELETE /api/cert/ca?
fingerprint=4deea7060d80babf1643b4e0f0104c82995075b8 //požadavek
{
  //odpověď
  "success" : true
}
---
{
  //odpověď
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "id",
    "description" : "certificate not found",
    "data": "cert_not_found"
  }
}
```

### 5.20.2 api cert user

Funkce **/api/cert/user** slouží ke správě uživatelských certifikátů.

Funkce je součástí služby **Systém API** a v případě použití autentizace je nutné, aby uživatel měl přiřazené privilegium **Systém - řízení**.

U této funkce lze použít metodu **GET**, **PUT** nebo **DELETE**. Metoda **PUT** vrací informace o certifikátech v zařízení. Metoda **PUT** slouží k nahrání certifikátů do zařízení.

Metoda **DELETE** slouží ke smazání certifikátu ze zařízení.

#### Metoda GET

Parametry požadavku **GET**:

Parametr	Popis
<b>id</b>	Volitelný parametr (string) identifikující uživatelský certifikát. Hodnota parametru <b>id</b> je přednastavené id nebo otisk (hash) certifikátu. Pokud není <b>id</b> zadáno, odpověď obsahuje úplný seznam všech uživatelských certifikátů v zařízení.

Odpověď je ve formátu **application/json** a může obsahovat následující parametry:

Parametr	Popis
<b>fingerprint</b>	Otisk (hash) certifikátu
<b>subject, issuer</b>	Informace o subjektu nebo vydavateli: Common Name (CN), Organization (O), Organization Unit (OU), Location (L), State (S), Country (C).
<b>id</b>	Hodnota ( <i>řetězec</i> ) dříve nastavené identifikace certifikátu.
<b>startdate</b>	Datum začátku platnosti certifikátu
<b>enddate</b>	Datum konce platnosti certifikátu
<b>protected</b>	Hodnota určující, že certifikát nemůže být smazán ( <i>true/false</i> ). Interní certifikáty s <b>id</b> začínajícím znakem # nemohou být smazány.
<b>systemUseOnly</b>	Hodnota určující, zda může daný certifikát uživatel zvolit jako certifikát pro jakoukoli službu. Pokud je hodnota <i>true</i> , certifikát se v seznamu pro výběr nezobrazí.

Příklad 2: Získání informací pro určitý certifikát identifikovaný otiskem

```
GET /api/cert/user?id=a164b11215a30f08603fd85c314327e274772b00 //požadavek
{
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint" : "a164b11215a30f08603fd85c314327e274772b00",
        "subject" : {
```

```

        "CN" : "00-0001-0205",
        "O" : "2N TELEKOMUNIKACE a.s.",
        "S" : "Czech Republic",
        "C" : "CZ"
    },
    "issuer" : {
        "CN" : "My2N Device Utility Certificate Authority",
        "O" : "2N TELEKOMUNIKACE a.s.",
        "S" : "Czech Republic",
        "C" : "CZ"
    },
    "startDate" : "2021-11-08T07:50:36Z",
    "endDate" : "2022-02-06T07:50:36Z",
    "protected" : false,
    "id" : "#my2n-utility",
    "systemUseOnly" : true
}
]
}
}

```

## Metoda PUT

Při opakovaném nahrání stejného certifikátu se původní certifikát přepíše.

Parametry požadavku **PUT**:

Parametr	Popis
<b>blob-cert</b>	Povinný parametr obsahující soubor s certifikátem ve formátu DER nebo PEM.
<b>blob-pk</b>	Povinný parametr soukromého klíče ve formátu DER nebo PEM
<b>password</b>	Nepovinný parametr obsahující heslo k nahraného klíče.
<b>id</b>	Volitelný parametr (string) identifikující uživatelský certifikát. Pokud je nahrán certifikát pod stejným <b>id</b> , původní certifikát s tímto <b>id</b> se přepíše.

Odpověď je ve formátu **application/json** a obsahuje otisk nahraného certifikátu

Parametr	Popis
<b>fingerprint</b>	Otisk (hash) nahraného certifikátu

Parametr	Popis
<b>replaced</b>	Otisk přepsaného certifikátu

## Příklad

```

PUT /api/cert/ca          //požadavek
{
  "success" : true,
  "result" : {
    "certificates" : [
      {
        "fingerprint": "9623fa25e414aa930ed22348a22d04a4c4fda26b"
      },
      {
        "fingerprint": "9623fa25e414aa930ed22348a22d04a4c4fda26b"
        "replaced": "9623fa25e414aa930ed22348a22d04a4c4fda26c"
      }
    ]
  }
}
Response
{
  "success" : false,
  "error" : {
    "code" : 12,
    "param" : "blob-cert",
    "description" : "invalid certificate",
    "data" : "invalid_cert"
  }
}

```

## Metoda DELETE

Parametry požadavku:

Para metr	Popis
<b>id</b>	Povinný parametr (string) identifikující uživatelský certifikát. Parametr <b>id</b> je přednastavené id nebo otisk (hash) certifikátu.

Odpověď je ve formátu **application/json**.

## Příklad

```
DELETE /api/cert/ca?fingerprint=4deea7060d80babf1643b4e0f0104c82995075b7
{
  "success": true
}
```

